MAZILESCU GEORGIAN

PROIECTAREA DIDACTICĂ

EXEMPLE DIN TEORIA GRAFURILOR

INFORMATICĂ

PROIECTAREA DIDACTICĂ

EXEMPLE DIN TEORIA GRAFURILOR

Autor: Mazilescu Georgian

ISBN 978-606-94763-3-8

Editura Evomind, 2020

https://evomind.org/

CUPRINS

INTRODUCERE

Lucrarea cuprinde aspecte metodice şi aplicaţii.

Fiecare capitol cuprinde o prezentare teoretică a informaţiei, iar la final aplicaţii practice la noţiunile prezentate.

Am ferma convingere că maniera în care este concepută lucrarea va duce la creşterea interesului pentru informatică şi mai ales pentru ***Teoria Grafurilor*** cu aplicabilitate mare în toate domeniile de activitate, arătând mai clar universalitatea aplicaţiilor ei.

1. OBIECTIVELE GENERALE ALE DISCIPLINEI INFORMATICĂ

Transformările societăţii româneşti din ultimii ani, dezvoltarea şi răspândirea informaticii, pătrunderea hardware-ului şi software-ului modem in viaţa economică, socială şi în învăţământ, impun o pregătire diversificată a tinerilor în acest domeniu. Învăţământul preuniversitar de informatică trebuie să asigure dobândirea, fie a unor cunoştinţe de informatică la nivel de cultură generală, necesare continuării studiului, fie a unor cunoştinţe având caracter aplicativ la un nivel mediu de profesionalism care să asigure posibilitatea găsirii unui loc de muncă.

Toţi tinerii trebuie să-şi asigure un minim de cunoştinţe de *tehnologia informaţiei*, necesare utilizării calculatoarelor în rezolvarea problemelor profesionale în diversele domenii ale vieţii economice. Indiferent dacă vor absolvi sau nu o instituţie de învăţământ superior, vor avea extrem de mult de câştigat dacă vor avea cunoştinţe de informatică, reuşind astfel să corespundă cerinţelor pe care locurile de muncă ale prezentului şi viitorului apropiat le vor ridica în faţa lor.

Obiectivele generale ale disciplinei informatică în învăţământul preuniversitar sunt:

a) Pornind de la faptul ca nu există domeniu de activitate unde să nu se prelucreze şi să nu se transmită informaţii atât în cadrul domeniului respectiv cât şi spre exteriorul lui, afirmăm că azi informaţia este foarte valoroasă ea trebuie stocată, prelucrată şi transmisă în condiţii care asigură corectitudine şi exactitate, deci la nivel profesional. Rezultă direct ca unul din obiectivele învăţământului de informatică trebuie să fie *asigurarea înţelegerii tuturor problemelor legate de informaţie şi de stocarea, prelucrarea, respectiv transmiterea ei*.

b) Dezvoltarea deprinderilor moderne de utilizator, adică pregătirea elevilor astfel încât să poată beneficia de lumea calculatoarelor, respectiv să poată folosi posibilităţile asigurate de cultura informatică, trebuie să stea în atenţia învăţământului preuniversitar. Aceasta presupune identificarea şi înţelegerea principalelor componente ale calculatorului, precum şi a funcţionării reţelelor de calculatoare. Elevii trebuie să

cunoască interfeţele utilizator ale sistemelor de operare şi ale celor mai răspândite utilitare, modul de instalare, exploatare şi utilizare a acestora, să dobândească deprinderi necesare cunoaşterii şi folosirii oricărui software nou, precum şi a versiunilor noi pentru cel existent.

c) *Dezvoltarea gândirii algoritmice* este un obiectiv la realizarea căruia informatica are o contribuţie esenţială şi eficientă. Asemenea matematicii, informatica dezvoltă gândirea în general şi are în şcoală, dar şi în viaţa de zi cu zi, un rol esenţial în procesul de învăţare, în formarea caracterului şi a personalităţii. Dar informatica (în plus faţă de matematică) dezvoltă gândirea algoritmică diferită de gândirea matematică (preponderent teoretică şi abstractă) prin faptul că obligă elevii să finalizeze rezolvări ale unor *aplicaţii practice concrete*. Această gândire nu se leagă doar de cunoştinţele de programare, ci şi de cunoştinţe referitoare la gestionarea bazelor de date, la utilizarea tabelatoarelor, editoarelor de texte, etc. Demonstrarea teoretică a existenţei unei soluţii pentru o problemă dată (ca în matematică) chiar dacă este importantă, nu este echivalentă cu rezolvarea problemei specificate. Este nevoie de dezvoltarea concrete de a *rezolva probleme*.

d) *Dezvoltarea deprinderilor necesare muncii individuale se realizează într-un proces firesc, în dialog cu calculatorul.* Acesta este un instrument care reacţionează imediat la încercările elevului şi care totodată nu îşi pierde răbdarea niciodată, oferă şansa unei învăţări conform ritmului propriu al fiecăruia, oferă posibilitatea lucrului diferenţiat cu elevi talentaţii sau cu cei care lucrează mai lent.

Informatica este esenţial legată de lucrul individual cu un calculator, deci dezvoltă într-un mod firesc *deprinderea de a lucra individual*. Din nefericire, aceasta poate conduce la formarea unor trăsături cum ar fi individualismul sau egoismul. Aici intervine esenţial rolul profesorilor: să încurajeze şi să organizeze activitate în grupuri.

e) Prin folosirea reţelelor de calculatoare a apărut posibilitatea unui schimb de informaţie între utilizatorii de calculatoare, mult mai eficient decât cel clasic (poştă, telefon, telex etc.). *Educarea elevilor în spiritul unei activităţi desfăşurate în grup, în colaborare, se finalizează prin predarea informaticii orientată pe proiecte*. Realizarea unor aplicaţii mai complexe impune lucrul în grup, modularizarea programului şi păstrarea contactelor cu ceilalţi membri ai grupului. în viaţa reală majoritatea activităţilor

nu se desfăşoară izolat, de aceea profesorii trebuie să stimuleze acest fel de activităţi. Proiectele trebuie să fie împărţite în activităţi repartizate pe individ, interconectate şi elevii trebuie învăţaţi să preia şi să transmită informaţii respectând anumite specificaţii. Evident, profesorul are rol de supervizor.

Obişnuirea elevilor cu diverse responsabilităţi, cu răspunderea privind finalizarea propriei munci şi asigurarea înlănţuirii unor elemente realizate în paralel, îi va pregăti pentru o activitate pe care cu siguranţă o vor întâlni în viitor.

f) Este important ca elevii să fie capabili să aleagă din instrumentarul existent pe cel de care au nevoie, *să identifice şi să folosească software-ul cel mai potrivit aplicaţiei* pe care o realizează.

Rezultă că trebuie să fie capabili *să analizeze problema* să descopere cerinţele şi să decidă ce software şi ce instrumente ale acestuia sunt cele mai utile. Pe de altă parte, pentru ca elevul să poate „alege" ceva, el trebuie să afle măcar de existenţa şi caracteristicile esenţiale mai multor tipuri de software.

g) Educarea elevilor, urmărind atent *dezvoltarea spiritului inventiv şi creator*, se realizează în mai multe sensuri în cadrul disciplinei informatică. Indiferent de conţinutul programului, sau al aplicaţiei, *ceea ce realizează elevul trebuie să funcţioneze*, trebuie să fie utilizabil; altfel spus, *trebuie să aibă toate calităţile unui produs finit*. Aceste cerinţe in informatică se concretizează prin:

- interfaţa prietenoasa;

- asigurarea *funcţionării* aplicaţiei *în mod inteligibil* chiar şi în cazul unui utilizator neautorizat, sau al unuia care nu cunoaşte aplicaţia;

- *fiabilitate*; aplicaţiile trebuie verificate şi testate;

- *performanţă*; analiza complexităţii (în cazul algoritmilor) şi a eficienţei (în cazul aplicaţiilor nealgoritmice) trebuie să devină obişnuinţă;

- portabilitate.

h) În liceu, informatica trebuie să pornească de la un nivel de bază, incluzând *Tehnologia Informaticii.* La acest nivel, nu putem spune că informatică este o disciplină izolată sau independentă. Un scop important este ca elevii **s***ă ştie să folosească tehnologia informaţiei* pentru a rezolva diverse probleme. Astfel, profesorii altor discipline pot prezenta sau solicita realizarea unor aplicaţii de tip software educaţional,

deci elevii ar trebui să ştie să utilizeze cunoştinţele dobândite la orele de informatică, realizând la rândul lor, instrumente noi, care se pot folosi în cadrul altor lecţii.

2. METODE DIDACTICE SPECIFICE ÎNVĂŢĂMÂNTULUI DE INFORMATICĂ

Metodele de învăţare sunt modalităţi de lucru de care profesorii şi elevii se folosesc în activitatea didactică. O metodă de învăţare este o structură de procedee, un program potrivit căruia se reglează acţiunile practice şi intelectuale întreprinse cu elevii în vederea realizării obiectivelor pedagogice.

Procedeul este un detaliu al metodei cu diferite funcţii în procesul predării-învăţării: înlătură obstacolele de înţelegere, stimulează şi motivează elevul, previne oboseala, monotonia etc.

Funcţiile metodelor de învăţare:

- *Funcţia de transmitere şi însuşire a cunoştinţelor*. Prin metodele de învăţare se asigură transferul experienţei, al valorilor de la societate la elev.
- *Funcţia formativ-educativă.* O metodă de învăţare se foloseşte în măsura în care aceasta contribuie la formarea anumitor abilităţi, capacităţi de ordin intelectual, afectiv sau psihomotor, contribuie la dezvoltarea anumitor trăsături de personalitate ale elevilor cu care profesorul lucrează.
- *Funcţia normativ-instrumentală.* Metodele orientează activitatea didactică, arată cum trebuie procedat pentru ca obiectivele activităţii să fie realizate, să se transpună în performanţe ale elevilor.
- *Funcţia motivaţională.* Prin metodele de învăţare folosite, profesorul stârneşte curiozitatea şi dezvoltă interesele de cunoaştere, de autodepăşire ale elevilor.

Metodele creează suportul motivaţional al învăţării.

Sarcinile didactice se realizează cu ajutorul metodelor, tehnicilor şi procedeelor didactice. Folosirea judicioasă a metodelor are o deosebită importanţă pentru reuşita activităţii de la catedră; pe de altă parte, conţinuturile fiecărei discipline şi obiectivele pe care şi le propune să le îndeplinească, pretind metode adecvate. Adoptarea şi nu adaptarea metodelor de predare ale unor discipline la alte discipline pot conduce la rezultate contradictorii.

Cert este că informatica poate adapta metode de predare de la alte discipline, dar adaptarea trebuie să se facă ţinând cont de:

- dinamica conţinuturilor şi particularităţile metodice ale predării disciplinei;
- individualizarea învăţării informaticii, ca disciplină deschisă şi dinamică;
- activismul care pretinde o participare prioritară conştientă a elevului la procesul de autoinstruire;
- studiul informaticii atât ca disciplină autonomă, cât şi ca instrument operaţional al altor discipline.

În cele ce urmează se vor analiza metodele utilizate în predarea informaticii: expunerea, conversaţia, problematizarea, modelarea, demonstrarea folosind materialul intuitiv, exerciţiul, învăţarea pe grupe mici, munca cu manualul, jocurile didactice, asaltul de idei (brainstorming), instruirea programată.

În tratarea acestor metode se vor urmări cu predilecţie particularităţile specifice predării disciplinelor de informatică şi în special, aplicaţiile practice de laborator şi contribuţia informaticii la realizarea obiectivelor didactice ale disciplinelor în învăţământul preuniversitar.

2.1. Expunerea sistematică a cunoştinţelor

Dintre formele pe care le îmbracă expunerea sistematică a cunoştinţelor (povestirea, prelegerea, descrierea, explicaţia, conversaţia etc.), opinăm că informatica utilizează cu precădere explicaţia. Elementele explicative domină procesul de instruire informatică, acestea fiind caracteristice atingerii unor obiective de referinţă care cuprind formarea de deprinderi şi abilităţi practice de utilizare a unor produse soft deseori complicate şi dominate de interfeţe „neprietenoase" faţă de utilizator (netransparente). Ceea ce conferă o accentuată notă de adaptabilitate este operativitatea impusă de aplicarea acestei metode prin alternarea expunerii cu demonstraţia practică, elevii fiind astfel scoşi din pasivitatea posturii de simpli receptori. Analogiile cu situaţii cunoscute fac din receptorul pasiv un participant activ la expunere. Expunerea nu se desfăşoară în condiţii perfect univoce, adică fără alternative şi reveniri, nici la disciplinele cărora

metoda le este caracteristică. La informatică, aceasta se întâmplă cu atât mai puţin. Elevul primeşte în condiţii univoce doar ceea ce i se comunică în funcţie de nivelul de cunoştinţe dobândit, de propriile-i presupuneri, de experienţa sa practică, de nivelul său de gândire, de înţelegerea codului de comunicaţie, ca să nu mai vorbim de oscilaţiile de atenţie. Profesorul trebuie să reproiecteze lecţia prin prisma posibilităţilor elevilor şi cu mijloacele lor de gândire. Accentul trebuie pus pe raţionament, prin argumentări temeinice, prin scoaterea în evidenţă a modului în care trebuie să gândească. Expunerea trebuie să fie însoţită de un control permanent al gradului de receptivitate al clasei, urmărindu-se mimica elevilor (edificatoare în special la elevii mici), satisfacţia înţelegerii lecţiei sau îngrijorarea şi neliniştea în cazul în care elevul a pierdut firul explicaţiei citindu-se pe faţa elevilor, întrebările, repetiţia, explicaţiile suplimentare, analogiile cu alte noţiuni cunoscute permit realizarea unui control permanent al receptivităţii la expunere. În informatică recurgem neapărat la metoda expunerii (explicaţiei) atunci când tema este complet nouă şi printr-o metodă activă nu se poate descoperi noutatea sau metoda activă este ineficientă din punctul de vedere al operativităţii. Astfel, este necesară această metodă pentru a înţelege noţiunea de algoritm (inclusiv exemplificările clasice), de structură de date (inclusiv exemplificările clasice), de structură de date (inclusiv modalităţile de reprezentare), de comandă, funcţie sau procedură standard (în legătură cu sistemul de operare sau mediul de programare ales), de raţionament (într-un spaţiu închis ales) şi chiar a modalităţii de prezentare şi introducere a unor programe utilitare, softuri de aplicaţie etc. În acest context, pentru reprezentarea comenzilor unui sistem de operare, a unui editor de texte (sau grafic), a altor softuri mai complicate (prevăzute de programa şcolară) se poate recurge la următoarele (sub)metode:

- Expunerea (la tablă, prin slide-uri pe retroproiector sau prin PowerPoint) cu „desenarea" meniurilor şi prezentarea funcţiilor fiecărei opţiuni, urmând ca elevul (prin aplicaţiile de laborator) să exerseze fiecare funcţie în parte, individual sau în grupe mici de lucru.
- Prezentarea meniurilor şi funcţiilor fiecărei opţiuni simultan cu exersarea acestora în cadrul orelor de aplicaţii practice de laborator.
- Prezentarea meniurilor şi funcţiilor fiecărei opţiuni simultan cu demonstrarea practică în momentul prezentării lor de către profesor, sarcina elevului fiind

numai aceea de a urmări şi reţine modul de executare a operaţiilor prezentate de profesor, urmând ca elevul să aplice cunoştinţele dobândite în cadrul orelor de laborator, în aplicaţii ample (integrate, de dorit, într-un mediu economic clar), care necesită utilizarea în mod repetat şi în situaţii diferite a funcţiilor fiecărei opţiuni din meniul discutat.

Fiecare dintre variantele de mai sus au avantajele şi dezavantajele lor. Prima variantă este cel mai des folosită deoarece, de regulă, profesorul nu are la dispoziţie un laborator şi pentru predare (iar aceasta se face cu întreaga clasă). Ea prezintă dezavantajul că elevul „nu vede pe viu" efectul executării fiecărei opţiuni (profesorul fiind nevoit în acest caz să-1 descrie în cuvinte), dinamica transformării şi efectul video al acestora fiind greu de redat în cuvinte. Singurul avantaj este cel al obţinerii de către elev a unui rezumat logic şi coerent după care se va ghida în timpul realizării unor aplicaţii practice. A doua variantă înlătură dezavantajul neobservării pe viu a efectului executării fiecărei opţiuni, dar atenţia elevului este îndreptată spre realizarea practică (simultan cu comunicarea modului de realizare a funcţiilor opţiunilor din meniuri). Astfel, o parte dintre funcţii sunt abordate prea „abrupt" sau sunt chiar omise, iar altele sunt exersate prea mult. La acest dezavantaj se adaugă şi reducerea randamentului prin faptul că profesorul trebuie să urmărească modul în care fiecare elev sau fiecare grupă aplică funcţia prezentată şi să intervină ori de câte ori un elev sau o grupă este în impas. În plus, unii elevi îşi formează mai repede deprinderea utilizării, iar alţii mai greu, primii fiind tentaţi să încerce între timp alte opţiuni (chiar neprezentate încă de către profesor), ceea ce creează disfuncţionalităţi în desfăşurarea lecţiei, aprecierea gradului de asimilare şi chiar formarea unor idei greşite de utilizare (datorate încercărilor individuale, necoordonate). Pe lângă acestea, se pierde din vedere şi realizarea unui rezumat sistematic al modului de optimizare, elevul fiind tentat să exerseze imediat funcţia şi uită să-şi noteze „în stil propriu" modul de utilizare a acesteia. Ultima variantă pare să cumuleze toate avantajele celor anterioare prin faptul că elevul urmăreşte şi reţine (neavând alte preocupări care să-i distragă atenţia) modul în care profesorul execută (corect) şi explică simultan, elevii putând nota tot ce acesta prezintă. Este o manieră de expunere ce înlătură formarea unor deprinderi greşite, mărind randamentul la predare şi asimilarea noilor cunoştinţe. Această variantă are însă şi un dezavantaj: necesitatea existării unei dotări speciale, care să permită observarea în bune condiţii, de către toţi elevii clasei, a ecranului calculatorului

pe care profesorul face demonstraţia. Utilizarea unui retroproiector sau a unui videoproiector are multe inconveniente (în afară de costul ridicat), printre care faptul că trebuie să existe anumite condiţii de mediu specifice în sala de clasă. De exemplu, pentru grupe mici poate fi folosit numai calculatorul ca atare, dacă elevii pot fi aşezaţi în preajma acestuia astfel încât fiecare să poată observa fără efort ecranul. Indiferent de conţinutul lecţiei, metoda expunerii nu se foloseşte singură decât foarte rar pe parcursul unei ore întregi, aceasta alternând cu alte metode de predare. Pe de altă parte, există o tendinţă accentuată a cadrelor didactice de a nu-şi propune aprioric folosirea cu precădere a nici unei metode, ceea ce este foarte dăunător.

2.2. Metoda conversaţiei

Metoda conversaţiei se referă la dialogul dintre profesor şi elev, în care profesorul nu trebuie să apară în rolul examinatorului permanent, ci în rolul unui colaborator care nu numai întreabă, ci şi răspunde la întrebările elevilor. Prin metoda conversaţiei se stimulează gândirea elevilor în vederea însuşirii, fixării şi sistematizării cunoştinţelor şi deprinderilor, în vederea dezvoltării spiritului de colaborare şi de echipă. Se asigură astfel o participare activă din partea elevilor, întrebările putând fi adresate (teoretic) în orice moment al lecţiei. Metoda conversaţiei este frecvent utilizată în învăţarea informaticii, ea implicând un dialog continuu între elev şi profesor, respectându-se anumite reguli elementare de colaborare constructivă care să nu determine diminuarea demersului didactic, ci să-l amplifice şi să-l consolideze. Conversaţia didactică poate îmbrăca forme diferite, în funcţie de anumite criterii. În funcţie de numărul de persoane, ea poate fi:

- Individuală. Se poartă între elev şi profesor.
- Colectivă sau frontală. Întrebările sunt adresate întregii clase, iar răspunsurile "vin" de la diferiţi elevi.

După obiectivele urmărite în diferite variante de lecţii, conversaţia poate fi:

- Introductivă. Aceasta este folosită în momentul captării atenţiei şi reactualizării cunoştinţelor asimilate anterior, pentru a trezi interesul pentru lecţia care urmează.

• Expozitivă. În timpul prezentării unei noi lecţii, ea poate trezi interesul pentru fixarea noilor cunoştinţe.

• Recapitulativă. Este utilizată atunci când se urmăreşte recapitularea şi generalizarea unor rezultate prezentate anterior.

• Evaluativă. Este indicată, desigur, pe parcursul procesului de evaluare şi verificare.

• Dezvoltată. Este destinată prezentării unui nou subiect, nu complet necunoscut.

Caracteristicile principale ale întrebărilor, indiferent de forma de conversaţie, impun precizie şi vizarea unui singur răspuns. De multe ori se pun întrebări vagi, care încep cu „Ce puteţi spune despre..." sau „Ce ştiţi despre...", care plasează elevul într-un dubiu total în legătură cu conţinutul răspunsului. Din aceeaşi gamă face parte şi celebrul îndemn de evaluare "Prezintă subiectul pe care-l cunoşti tu cel mai bine". Nu este normal ca întrebarea să conţină răspunsul sau să ceară un răspuns prin „da" sau „nu". Ea contribuie la dezvoltarea gândirii. De asemenea, răspunsurile acceptate trebuie să fie corecte, complete, exprimate în termeni precişi, să oglindească o înţelegere efectivă a problemei abordate. Discuţiile au şi rolul de a corecta greşelile din răspuns. Identificarea cauzei, eliminarea greşelii, cât şi posibilitatea reapariţiei ei sunt foarte importante. Conversaţia are un rol primordial prin faptul că ajută la formarea limbajului informatic, la dezvoltarea raţionamentului logic şi a gândirii elevului. Dificultăţile pe care elevul le întâmpină în formarea limbajului de specialitate pot lăsa urme în plan afectiv, repercutându-se asupra dezvoltării lui intelectuale. De aceea se impune o analiză amănunţită a cauzelor acestor dificultăţi, iar scoaterea lor în evidenţă trebuie relevată prin examinări (scrise, orale, reprezentări schematice, utilizarea simbolurilor specifice). A fi la curent cu dificultăţile de limbaj pe care le au elevii la anumite vârste şcolare şi la un anumit stadiu de însuşire al disciplinei înseamnă în primul rând să nu se abuzeze de termeni de specialitate (înlocuindu-i cu termeni sinonimi din vocabularul curent sau explicându-le sensul, dacă un alt înţeles al termenului este accesibil). Dificultatea formării vocabularului de specialitate constă şi în faptul că aceste cuvinte noi sunt introduse în acelaşi timp cu introducerea noţiunilor noi, ceea ce face ca îmbogăţirea limbajului informatic să se facă simultan cu dezvoltarea şi formarea gândirii informatice. Stăpânirea limbajului se reflectă în rezolvarea problemelor şi înţelegerea textelor şi documentaţiilor de specialitate. Nestăpânirea acestuia provoacă inhibiţie, imposibilitatea

comunicării sau chiar o comunicare şi o înţelegere defectuoasă, făcându-1 pe elev timid, incoerent sau chiar ridicol în exprimare. Această metodă mai are şi următoarele subdirecţii:

Euristică. Nu există reguli precise, se bazează doar pe întrebare/răspuns, în funcţie de evoluţia concretă a dialogului.

Tip dezbatere. Se realizează un schimb de păreri în care este implicat un anumit colectiv. Ar fi bine să fie trase şi nişte concluzii care să nu aibă doar un rol istoric.

Catehetică. Aceasta impune efectuarea unor teste care implică memoria.

Este clar că o conversaţie se face prin întrebări. În plus, acestea trebuie să satisfacă următoarele condiţii (unele dintre ele rezultă din ceea ce am amintit mai înainte):

Să fie precise (vizând un singur răspuns)

Să nu conţină răspunsul şi să aibă un rol instructiv.

Să stimuleze gândirea şi capacitatea de creativitate a elevilor („De ce?", „Din ce cauză?" etc.)

Să fie formulate prin enunţuri variate şi „atrăgătoare".

Să se adreseze întregului colectiv vizat.

• Să conţină întrebări ajutătoare atunci când răspunsul este eronat sau parţial. Răspunsurile acceptate trebuie să fie nu numai corecte, ci şi exprimate în termeni precişi şi să oglindească un anumit nivel de înţelegere. Răspunsurile eronate trebuie corectate imediat, prin discuţii individuale. Cadrul didactic trebuie să dirijeze conversaţia astfel încât ideile să fie bine conturate înainte de a trece la altele, în timp ce lecţia îşi menţine caracterul unitar. În ceea ce priveşte informatica, recomandăm şi utilizarea unor instrumente ajutătoare, ca de exemplu introducerea/exprimarea noţiunilor printr-un „limbaj de programare" (scris/oral) care să implice utilizarea eficientă a simbolurilor (în afară de latura didactică propriu-zisă), ceea ce înseamnă separarea clară a sintaxei de semantică.

2.3. Problematizarea şi învăţarea prin descoperire

Predarea şi învăţarea prin problematizare şi descoperire presupun utilizarea unor tehnici care să producă elevului conştientizarea „conflictului" dintre informaţia dobândită şi o nouă informaţie, determinându-l să acţioneze în direcţia lichidării acestuia prin descoperirea unor (noi) proprietăţi ale fenomenului studiat. Pedagogic vorbind, conflictele se mai numesc şi situaţii-problemă, putând fi de cel puţin două tipuri:

Contradicţii între posibilităţile existente ale elevului (nivelul intelectual şi de pregătire) şi cerinţele, situaţiile în care este pus de noua problemă. Aceste conflicte se datorează imposibilităţii elevului de a selecta dintre cunoştinţele sale anterioare pe cele potrivite cu valoarea operaţională de aplicabilitate a viitorului.

Incapacitatea elevului de a integra noţiunile selectate într-un sistem, în acelaşi timp cu conştientizarea faptului că sistemul este pe moment ineficient operaţional (lucru care poate fi remediat doar prin completarea informaţiei de bază).

Întrebările frontale sau individuale utilizate în etapa de pregătire a introducerii unei noţiuni, a prezentării unui domeniu nou, întrebări care se adresează capacităţii de reacţionare a individului, pot genera noi situaţii conflictuale de tipul menţionat anterior. Pe cât posibil, cadrul didactic trebuie să gestioneze el însuşi apariţia situaţiilor-problemă. La modul ideal, ele trebuie să apară de la sine în mintea elevului. Relativ la condiţiile pedagogice ale acestor situaţii conflictuale generate de anumite probleme practice putem spune că problemele trebuie să aibă un sens precis şi să fie enunţate într-un moment „optim" al lecţiei. Ele trebuie să înglobeze cunoştinţe anterior însuşite de elev, să le trezească interesul, să le solicite un anumit efort mental creator. Există părerea că rezolvarea problemei poate fi privită ca un proces prin care elevul descoperă că o combinaţie de reguli învăţate anterior se poate aplica pentru găsirea soluţiei unei noi situaţii conflictuale. În acest sens se pot evidenţia următoarele etape în rezolvarea problemei:

- Prezentarea problemei (verbal, scris, grafic etc.)
- Definirea problemei de către elev în sensul distingerii caracteristicilor esenţiale ale situaţiei, însuşirii enunţului, găsirii legăturii între date, informaţii etc.

- Formularea de către elev a anumitor criterii, ipoteze care pot fi aplicate în vederea găsirii unei soluţii. Verificarea succesivă a unor asemenea ipoteze, eventual şi a altora noi şi găsirea efectivă a unei soluţii (sau a tuturor).

Desigur că în contextul de mai sus expresiile situaţie conflictuală, problemă, rezolvare de problemă se referă la probleme şi soluţii noi, necunoscute încă de elev, şi nu la ceva de tipul substituirii de valori numerice în expresii date, execuţia unui program dat pentru nişte valori de intrare etc. Utilizarea în predare a acestei metode este totdeauna utilă în momentul în care se găseşte şi rezolvarea conflictului.

Descoperirea apare ca o întregire a problematizării. Se pot pune astfel în evidenţă trei modalităţi principale de învăţare prin problematizare şi descoperire (clasificarea făcându-se după tipul de raţionament folosit):

Modalitatea inductivă;

Modalitatea deductivă;

Modalitatea prin analogie.

În primul caz este vorba de generalizări. Elevul trebuie să-şi dezvolte propria cale de învăţare, care să nu contrazică lucrurile în care deja „crede", prin folosirea unor mijloace tehnice şi resurse informaţionale personale. În al doilea caz se foloseşte logica sau, mai exact, sistemele deductive (ca metodă de raţionament). Putem deriva cunoştinţe noi din cunoştinţe vechi (cu ajutorul unor reguli de inferenţă specifice). În ultimul caz, se încurajează folosirea unei experienţe anterioare nu numai dintr-un domeniu conex, ci chiar din domenii total diferite.

Problematizarea are astfel inferenţe cu conversaţia, întrebările individuale sau frontale adresate gândirii, raţionamentului născând situaţii conflictuale. Generarea situaţiilor-problemă trebuie produsă astfel încât întrebările să apară în mintea elevului, fără ca acestea să fie puse de către profesor. După cum am mai precizat, ca disciplină cu caracter formativ, informatica îşi propune formarea unei gândiri algoritmice, sistematice şi riguroase, care să promoveze creativitatea, să stimuleze imaginaţia şi să combată rutina. Chiar dacă aparent travaliul informatic se sprijină pe anumite şabloane, acestea reprezintă numai tendinţe utile de standardizare. Procesele care izvorăsc din situaţii reale, care implică folosirea calculatorului în rezolvarea unor probleme aparţinând diferitelor sfere ale vieţii de zi cu zi, analiza acestor probleme, alegerea structurilor de date pe care

se mulează informaţia oferită de mediul înconjurător, paşii algoritmilor şi programarea în sine determină folosirea metodei problematizării, iar aplicarea acestei metode necesită formarea unor deprinderi ce nu se obţin decât printr-un exerciţiu îndelungat. Rezolvarea de probleme, ceva curent în învăţarea informaticii, poate fi privită ca un proces prin care elevul descoperă că o altă combinaţie de reguli învăţate anterior conduc la rezolvarea unei noi situaţii problematice. Formularea de probleme de către elevii înşişi constituie forme ale creativităţii şi presupune că elevii şi-au format deprinderi intelectuale eficiente din punctul de vedere al generalizării şi aplicabilităţii (orice soluţie generează o nouă problemă). Problemele propuse pot fi inspirate din viaţa cotidiană, din cunoştinţele dobândite prin studiul altor discipline, din generalizarea unor probleme de informatică rezolvate anterior, probleme de perspicacitate, jocuri etc. Problematizarea şi descoperirea fac parte din metodele formativ-participative, care solicită gândirea creatoare a elevului, îi pun la încercare voinţa, îi dezvoltă imaginaţia, îi îmbogăţesc experienţa. În lecţiile în care se aplică aceste metode profesorul alege problemele, le formulează, dirijează învăţarea şi controlează munca depusă de elev în toate etapele activităţii sale. Această metodă este caracteristică, de exemplu, unor lecţii de aplicaţii practice de laborator, metoda învăţării prin descoperire fiind frecvent aplicată în momentul în care este necesară folosirea programelor utilitare, a softurilor de aplicaţie etc. Utilitarele se abordează în funcţie de problemele concrete care urmează a fi rezolvate. Obiectivul imediat este cunoaşterea şi exploatarea produsului, nu îmbunătăţirea lui. Concentrarea atenţiei va fi dirijată spre rezolvarea problemei şi nu asupra analizei facilităţilor şi lipsurilor produsului de software. Cu siguranţă, în acest caz este deosebit de importantă experienţa dobândită, cunoştinţele şi deprinderile formate în alte situaţii similare de învăţare: lucrul cu meniuri, funcţii comune mai multor utilitare, cunoaşterea structurilor de date, dexteritatea în tehnoredactare etc. Cunoaşterea facilităţilor produsului soft se face în momentul ivirii necesităţii exploatării acestuia şi nu printr-o prezentare a lui ca o înşiruire mai mult sau mai puţin sistematică şi completă de funcţii sau facilităţi. Bineînţeles că este obligatorie o prezentare generală a utilitarului. În contextul altor produse similare, trebuie concepută o viziune de ansamblu din care să se desprindă caracteristicile dominante ale utilitarelor din clasa respectivă şi să se prezinte

particularităţile specifice produsului, cu îmbunătăţiri faţă de versiunile anterioare şi perspective de dezvoltare pentru cele viitoare.

Ca informaticieni, ne interesează (în acest context) ceea ce numim rezolvarea problemelor (problem solving). Îndemânările dobândite în legătură cu acest subiect depind în primul rând de cunoştinţele specifice acumulate, dar din punctul de vedere al psihologiei există acordul că se pot căpăta şi „îndemânări generale". Procesul cognitiv în ansamblu este foarte complicat, numai pentru explicarea coerentă a acestuia fiind necesară o întreagă carte. Vom sublinia doar câteva elemente-cheie şi direcţii principale pentru abordarea rezolvării unor probleme. Astfel, când dorim să rezolvăm o problemă cu ajutorul calculatorului, presupunând că enunţul este „acceptat", trebuie să ne întrebăm în primul rând:

Ce ştim în legătură cu domeniul implicat?

Cum sunt apreciate pe piaţă rezultatele?

Care strategii generale sunt aplicabile?

Care sunt motivaţiile suplimentare?

După ce problema a fost enunţată şi sunt furnizate anumite indicaţii suplimentare, putem trece la alegerea strategiei concrete de rezolvare. Aceasta trebuie să fie selectată după un anumit plan, să permită un anumit tip de verificare şi generalizare. De asemenea, trebuie avute în vedere metode sau metodologii prin care să se interzică anumite „ramuri" şi să se permită explorarea de direcţii colaterale. Una dintre strategiile generale poate fi următoarea:

Pot să rezolv problema (am cunoştinţele necesare).

Definesc în mod (semi)formal.

Caut informaţiile suplimentare astfel încât să am o definiţie formală concretă (eventual, chiar într-un limbaj de programare concret).

Fac planul de implementare.

Îl execut (scriu „programele" şi le „rulez").

Verific faptul că ceea ce am făcut este „corect".

Generalizez (la alte cazuri, la alte probleme).

Peste tot, cunoaşterea măcar a unei părţi din logica formală este indispensabilă.

2.4. Modelarea

Modelarea ca metodă pedagogică poate fi descrisă ca fiind un mod de lucru prin care gândirea elevului este condusă la descoperirea adevărului, folosind un aşa-numit model şi utilizându-se raţionamentul prin analogie. Modelul şi metoda în sine nu presupun o asemănare perfectă cu cazurile reale iniţial specificate, ci numai cu o analogie rezonabilă. Ea constă în construirea unui sistem s1 a cărui descriere coincide cu descrierea sistemului original s; până la un anumit punct, s1 poate avea o natură diferită şi este în general mai simplificat şi formalizat. Ideea este că, investigând sistemul s1 prin metode specifice legate de o anumită temă de lecţie, se pot găsi noi soluţii, care apoi pot fi translatate în concluzii asupra sistemului de bază. Modelarea are o mare valoare euristică colaterală, prin utilizarea ei putându-se dezvolta spiritul de observaţie, capacitatea de analiză şi sinteză, creativitatea. Ideea ar fi să putem determina elevii să descopere singuri modelul. Astfel elevul se obişnuieşte să creeze noi probleme ce trebuie rezolvate, să adapteze algoritmi cunoscuţi la situaţii noi etc. Realitatea înconjurătoare este percepută şi înţeleasă pe baza unor modele deja cunoscute. Dezvoltarea deprinderilor de modelare, obişnuirea elevilor cu gândirea logică se realizează prin prezentarea exactă şi clară a modelelor şi prin transparenţa particularizărilor. Un exemplu edificator îl constituie învăţarea metodelor de elaborare a algoritmilor. Necesitatea unor formalizări se impune prin rigoarea modului de abordare a problemei, prin sistematizarea organizării informaţiei de intrare, a exactităţii proiectării prelucrării şi prin standardizarea ieşirii. Formalizarea necesită cunoştinţe dobândite în studiul altor discipline, fundamentate teoretic, iar accesibilitatea formalizării este condiţionată de factori specifici nivelului de cunoştinţe dobândite anterior, de categoria de vârstă, de capacitatea de asimilare. Abordarea ponderată a acestor aspecte conduce la dezvoltarea deprinderilor de abstractizare, a gândirii algoritmice şi sistemice. Utilizarea modelelor în realizarea algoritmilor presupune stabilirea unor analogii şi în organizarea datelor de intrare. Învăţarea algoritmilor este legată de cunoaşterea modului de organizare a datelor, de cunoaşterea profundă a structurilor de date posibile a fi prelucrate uşor de către calculator. Etapa cea mai importantă este cea a descoperirii algoritmului, urmată de stabilirea modului de organizare a datelor, dar importanţa acestui ultim aspect este

esenţială în determinarea performanţelor produsului program care implementează algoritmul. Modelarea (ca metodă pedagogică) este definită ca un mod de lucru prin care gândirea elevului este condusă la descoperirea adevărului cu ajutorul modelului, graţie raţionamentului prin analogie. Modelarea similară constă în realizarea unui sistem de aceeaşi natură cu originalul care să permită evidenţierea trăsăturilor esenţiale ale originalului. O gamă variată de probleme sunt rezolvate prin metoda Backtracking. Pentru implementarea într-un limbaj de programare a unui algoritm implementat prin Backtracking, elevul are nevoie de un model reprezentat de un program, cum ar fi cel de generare a permutărilor sau de rezolvare a problemei celor opt dame şi, prin mici modificări, el poate obţine multe alte programe care implementează algoritmi ce rezolvă probleme clasice, cum ar fi: generarea aranjamentelor, combinărilor, problema parantezelor, partiţiile unei mulţimi, problema celor opt turnuri etc. Similar se procedează în rezolvarea problemelor care necesită utilizarea stivelor sau cozilor, folosind operaţiile elementare cu elementele acestor structuri dinamice elementare. Modelarea analogică nu presupune o asemănare perfectă cu originalul, ci numai folosirea unei analogii. Modelele cunoaşterii în procesul modelării sunt:

Trecerea de la original la model;

Transformarea modelului sau experimentarea pe model;

Transferul pe original a rezultatelor obţinute pe model;

Verificarea experimentală pe original a proprietăţilor obţinute pe model.

Trecerea de la original la model se face prin simplificare. Se impune ca simplificarea să nu fie exagerată, pentru a nu se omite trăsăturile esenţiale ale originalului. Totodată, trebuie să nu se scape din vedere că valoarea modelului va fi apreciată prin prisma eficacităţii lui, adică a posibilităţilor pe care le oferă pentru atingerea scopului şi că noile informaţii obţinute pe baza modelului vor fi transferate cu grijă asupra originalului, având în vedere diferenţa dintre model şi original. Modelul devine astfel purtătorul unei semnificaţii, informaţii care poate fi exprimat printr-un suport material sau ideal. O clasificare a modelelor după natura suportului sub care se vehiculează informaţia poate fi:

Modele materiale, care au suport concret şi care se folosesc foarte puţin în învăţarea informaticii: folosirea unei table de şah în rezolvarea problemei celor opt dame

determină o rapidă înţelegere a mecanismului metodei Backtracking; utilizarea unei stive de monede de dimensiuni diferite pentru înţelegerea rezolvării problemei turnurilor din Hanoi. Nu trebuie exclusă posibilitatea învăţării direct pe obiectul de studiu, caz întâlnit (şi recomandat) în studiul structurii şi arhitecturii sistemelor de calcul şi a conexiunilor între ele, în contextul funcţionalităţii ca un ansamblu (sistem), este esenţială.
Modele ideale (virtuale), care se exprimă prin imagini, sisteme de simboluri sau semne convenţionale.

Învăţarea informaticii prin modelare presupune două etape. Într-o primă etapă, învăţarea se va face pe baza modelelor construite „de profesori", etapă în care se vor analiza trăsăturile modelului şi compararea lui cu originalul. Pentru a reliefa condiţiile pe care trebuie să le îndeplinească modelul, se vor da şi contraexemple. În a doua etapă, elevii vor fi deprinşi să construiască singuri modele. Importanţa descoperirii modelului de către elev constă în faptul că elevul este obişnuit să reprezinte într-o formă standard condiţiile impuse de problemă şi-şi adânceşte convingerea că informatica este un domeniu în care rezultatele pozitive se obţin doar printr-o înlănţuire logică de raţionamente. Folosirea modelelor nu înseamnă impunerea unor metode care trebuie reţinute şi aplicate orbeşte. Se va pune accentul pe înţelegerea paşilor unui algoritm şi se va încuraja prezentarea oricăror metode care exclud modelul şi care se impun prin eleganţă şi eficienţă. Elevii vor fi încurajaţi să-şi dezvolte şi să-şi prezinte ideile proprii, contribuind astfel la creşterea încrederii în posibilităţile lor, în valoarea ideilor lor. Ei nu trebuie să fie obligaţi să reproducă ideile altora, să aştepte ca totul să fie prezentat de profesor, să asimileze reţete, ci să descopere metode noi, să le prezinte, analizeze şi perfecţioneze printr-o comunicare continuă şi constructivă. Folosirea modelelor în învăţare deschide pentru informatică o impresionantă arie de aplicabilitate (inclusiv utilizarea ei în predarea altor discipline, de la artele plastice la cele mai diverse domenii ale tehnicii).

2.5. Exemplificarea sau demonstrarea materialului intuitiv

Prin exemplificare sau demonstraţie, în acest caz, înţelegem prezentarea sistematizată şi organizată a unor obiecte, procese, experimente, cu scopul de a uşura înţelegerea intuitivă şi executarea corectă a unor activităţi programate. Cuvântul intuiţie din titlu înseamnă utilizarea oricărui raţionament inductiv, în contextul temei şi bagajului de cunoştinţe ale elevului. Utilizarea intuiţiei împreună cu exemplificarea necesară poate implica folosirea a diverse modalităţi şi tehnici didactice datorită diversităţii materialului de studiu. Exemplificarea sau demonstrarea materialului intuitiv presupune utilizarea obiectelor reale, cum ar fi: material grafic (planşe, scheme), retroproiector/videoproiector şi material pretipărit, calculator (imagini grafice, multimedia, Power Point). În acest context, putem spune că: „Prin demonstrarea materialului intuitiv se înţelege prezentarea sistematică şi organizată a unor obiecte, procese etc. sau producerea unor experienţe, fenomene în faţa elevilor, cu scopul de a uşura înţelegerea şi executarea corectă a unor activităţi". Un rol deosebit îl joacă astfel intuiţia (intuiţia este o experienţă mentală; înseamnă o simplă observare şi notare a unor fapte; intuiţia poate fi asimilată cu un raţionament de tip inductiv). Intuiţia realizează corelaţia dintre imagine şi cuvânt, fiind atât sursă de cunoştinţe, cât şi mijloc de verificare. Informatica nu poate fi desprinsă decât artificial de bazele ei intuitive şi de extinderea ei în realitatea cotidiană. Convertirea principiului intuiţiei în metoda demonstraţiei se realizează în funcţie de materialul intuitiv: machete, grafică, film didactic, televiziune şcolară, software-uri de învăţare.

Ţinând cont de eficienţa transmiterii informaţiei prin mijloacele vizuale (inclusiv Internet) şi de orientarea cu predilecţie spre mijloacele de orientare rapidă care solicită atât memoria vizuală, cât şi cea auditivă şi formarea involuntară a unui public consumator de informaţie audio-video, o orientare a metodelor şi procedeelor didactice în vederea exploatării acestei stări de lucruri creează un avantaj aparte procesului instructiv-educativ. Crearea unor filme (casete video) didactice care să urmărească cu exactitate programa şcolară creează facilităţi de predare multor discipline şi ar permite elevului să poată revizualiza predarea lecţiei. Aceasta ar putea elimina ambiguităţile sau golurile

create de momentele de neatenţie din timpul predării şi ar constitui un veritabil profesor la purtător al elevului. Este evident că acest mijloc didactic nu poate înlocui (nici măcar suplini) exerciţiul individual şi nici prezenţa efectivă a cadrului didactic. Efortul profesorului este însă cu totul special. Nu este suficient ca un elev să vadă un material, el trebuie învăţat să vadă. Pentru că în acest moment ar trebui să aducem în discuţie euristicile şi încurajarea creativităţii. Se pot pune în evidenţă chiar euristici pentru dezvoltarea creativităţii:

încercaţi să aveţi cât mai multe idei. Cu cât sunt mai multe, cu atât puteţi selecta câteva „bune".

„Inversaţi" (reformulaţi, reiteraţi, puneţi într-un alt context) problema.

„Ghiciţi" o soluţie la întâmplare.

Gândiţi-vă la ceva distractiv, apropos de utilizările posibile ale rezolvării.

Gândiţi-vă la probleme similare şi la soluţiile acestora, chiar în contexte diferite.

Concepeţi o listă generală „explicativă" de cuvinte-cheie, proprietăţi utile, stimulente, ş.a.m.d. care au cât de cât legătură cu tema în cauză.

2.6. Metoda exerciţiului

La modul general, exerciţiile pot fi privite ca acţiuni concrete efectuate conştient şi repetat în scopul dobândirii unor priceperi şi deprinderi (mai rar cunoştinţe) noi, pentru a uşura anumite activităţi şi a contribui la dezvoltarea unor aptitudini. Avantajele metodei exerciţiului sunt:

-se poate forma o gândire productivă, creatoare, cu implicaţie financiară;

-se oferă posibilitatea câştigării unei anumite independenţe;

-se oferă posibilitatea iniţierii unui dialog-conversaţie cu obiective precise asupra unor metode şi soluţii;

-se activează atitudinea critică şi poate creşte discernământul elevilor în privinţa celor mai bune metode de lucru;

-se oferă profesorului o anumită posibilitate de a analiza şi evalua activitatea sau performanţele generale ale unui elev.

Condiţia primordială de reuşită este dată în principal de selecţia corespunzătoare a problemelor sau exerciţiilor, precum şi de activitatea de îndrumare-proiectare. Prin urmare, exerciţiile sunt acţiuni efectuate în mod conştient şi repetat de către elev cu scopul dobândirii unor priceperi şi deprinderi şi chiar cunoştinţe noi, pentru a uşura alte activităţi şi a contribui la dezvoltarea altor aptitudini. Însuşirea cunoştinţelor de informatică este organic legată de exersarea utilizării softului de aplicaţie, de rezolvarea unor probleme de programare etc. Nu există lecţie în care să nu se aplice această metodă. Alte avantaje sunt concretizate în rezultatele aplicării ei: formează o gândire productivă, oferă posibilitatea muncii independente, oferă posibilitatea analizei diverselor metode şi soluţii de rezolvare a problemelor, activează simţul critic şi autocritic şi îi învaţă pe elevi să-şi aprecieze rezultatele şi metodele de lucru, oferă posibilitatea depistării şi eliminării erorilor.

Este clar că metoda nu contribuie numai la formarea priceperilor şi deprinderilor de lucru cu calculatorul, ci contribuie substanţial la formarea unui comportament flexibil şi operant. Pentru profesor, alegerea, formularea şi rezolvarea problemelor şi apoi exploatarea rezultatelor obţinute constituie o sarcină de importanţă deosebită. Alegerea problemelor este condiţionată de programa analitică, succesiunea prezentării noţiunilor în manuale, metodele de rezolvare ce pot fi folosite şi de elevii cărora li se adresează. Formularea problemelor trebuie să ţină cont de noţiunile cunoscute de elevi, să fie clară, concisă şi să folosească limbajul de specialitate numai în măsura în care este cunoscut elevilor. Rezolvarea trebuie să aibă în vedere obţinerea rezultatelor pe căi clare şi uşor de verificat, reţinerea tipurilor de raţionamente folosite, deschiderea perspectivei pentru rezolvarea unor probleme analoage sau mai complexe. Folosirea rezultatelor obţinute trebuie să vizeze lămurirea conţinutului activ în cunoaşterea noţiunilor învăţate şi adâncirea semnificaţiei lor, asimilarea metodelor de rezolvare şi aplicarea lor la rezolvarea altor probleme. Utilizarea pe scară largă a acestei metode a condus la clasificarea exerciţiilor şi problemelor în funcţie de aportul capacităţilor intelectuale necesare rezolvării lor. În subsecţiunile care urmează insistăm asupra unor particularizări.

2.7. Probleme care permit însuşirea unor noţiuni

Specifice informaticii sunt problemele al căror grad de dificultate creşte treptat, odată cu formarea şi asimilarea noţiunii, fiecare nouă problemă aducând un plus de dificultate. În rezolvarea unei probleme de programare este necesar să se ţină cont de următoarele etape:

Analiza iniţială a problemei prin care se stabileşte formatul şi natura datelor de intrare, intervalele de variaţie a datelor de intrare, variabilele de lucru (date intermediare), precum şi formatul şi intervalele de variaţie a datelor de ieşire. Tot în această etapă se va stabili un algoritm (plan) de rezolvare, exprimat, eventual, în limbaj natural, pe baza căruia se va permite fiecărui elev să lucreze independent.

Rezolvarea propriu-zisă a problemei este etapa în care se realizează transpunerea într-un limbaj de programare a algoritmului stabilit în prima etapă. În prealabil, algoritmul este reprezentat într-una dintre formele cunoscute, se stabilesc variabilele de lucru, forma lor de alocare, prelucrările ce vor avea loc, apoi se trece la implementarea în limbajul dorit. Dacă rezolvarea se poate face pe mai multe căi, trebuie să se sublinieze, dacă este posibil, calea optimă.

Verificarea soluţiei sau soluţiilor obţinute va permite elevului să-şi dea seama dacă soluţia obţinută este cea corectă. În această etapă intervine profesorul cu seturi de date de test care să cuprindă, dacă este posibil, majoritatea (dacă nu toate) cazurilor ridicate de problemă şi în special cazurile critice, la limită, ale datelor de intrare.

Aceste date cuprind în esenţă: însuşirea enunţului, discutarea problemei şi stabilirea algoritmului de rezolvare, rezolvarea propriu-zisă, verificarea soluţiilor. Ele se pot modifica după natura problemelor. Acolo unde problema permite mai multe căi de rezolvare, profesorul analizează toate aceste căi şi le selectează pe cele mai importante, propunându-le spre rezolvare pe grupe, comparând rezultatele, avantajele şi dezavantajele fiecărei metode în parte. Se va evidenţia în mod obligatoriu cea mai bună soluţie.

O posibilă clasificare a problemelor/exerciţiilor (relativ la capacităţile intelectuale pentru rezolvare) ar fi:

- Exerciţii de recunoaştere a unor noţiuni (unitate curentă de I/E, unitate de disc, memorie internă, comandă externă, programe executabile de tip .com sau .exe, HTTP-uri, telnet etc.)
- Exerciţii aplicative (programe pentru transcrierea unor formule, pseudocoduri)

Aceste două clase de exerciţii sunt recomandate în special pentru fixarea unor cunoştinţe deja predate. În acest context poate fi utilă o complicare graduală a enunţului iniţial, urmărindu-se memorarea mai bună a formulei sau a ideii algoritmului, cum ar fi: încadrarea acestuia într-un eventual alt tip de probleme cunoscute, complicarea lui în mod progresiv în vederea utilizării sale în alte situaţii, prezentarea unor cazuri-limită care pot conduce la rezultate eronate.

- Exerciţii grafice - planşe, vizualizări
- Exerciţii complexe - presupun o analiză mai detaliată a problemei în ansamblu şi implică descompunerea problemei în subprobleme, succesiv, până în momentul în care rezolvarea subproblemelor elementare este cunoscută.

În rezolvarea exerciţiilor este importantă crearea posibilităţii îndeplinirii unei independenţe (individual, grup, echipă). Pentru formarea unor priceperi sau abilităţi legate de munca independentă, se poate utiliza şi aşa-numita formulă a exerciţiilor comentate. Aceasta constă în rezolvarea exerciţiilor de către toţi elevii, în timp ce un elev desemnat explică permanent rezultatele obţinute. Nu este nevoie ca această explicaţie să fie utilizată pe calculator. Profesorul poate în orice moment să invite oricare alt elev pentru continuarea explicaţiei (în acest fel, metoda este deosebit de activă). Discuţiile suplimentare sunt obligatorii în acest caz. Se vor evidenţia permanent avantajele şi dezavantajele rezolvărilor propuse, alte metode de rezolvare, idei privind utilizarea acestor rezolvări în lecţiile următoare, particularizări ale lor în lecţiile anterioare.

2.8. Metoda învăţării în grupe mici

Activitatea de învăţare pe grupe mici se defineşte ca o metodă în care sarcinile sunt executate de grupuri de elevi, grupuri care sunt câteodată autoconstituite şi care se autodirijează. Activitatea în informatică se desfăşoară în general în echipă, travaliul individual fiind o componentă a muncii corelate din cadrul unui grup de lucru. Tehnicile de organizare a muncii în unităţile de informatică evidenţiază ca o formă de organizare echipa programatorului-şef, echipă în care fiecare membru are sarcini bine stabilite (de analiză, programare, implementare, exploatare), sarcini corelate între ele. Este normal ca şi activitatea didactică să recurgă la metode de învăţare colectivă, fără a neglija însă munca individuală, ci doar privind-o pe aceasta ca o componentă a muncii în echipă. Profesorii recunosc, în general, eficacitatea unei asemenea organizări a activităţii didactice şi o integrează în arsenalul metodic al predării disciplinei. Criteriile de formare a grupelor sunt în funcţie de obiectivele urmărite (însuşirea de noi cunoştinţe, rezolvare de probleme etc.): grupuri omogene, formate din elevi cu acelaşi nivel de cunoştinţe; grupuri eterogene, formate din elevi de toate categoriile (foarte buni, buni şi slabi), dar în proporţii apropiate; grupuri formate pe criterii afective (prieteni, vecini de bancă). Numărul elevilor dintr-un grup poate varia de la 2 la 10, dar cele mai potrivite grupuri sunt cele formate din 4-6 elevi. La lecţiile de aplicaţii practice de laborator, grupurile de lucru formate din 4 elevi, care dispun de două calculatoare par a fi cele mai eficiente. Grupuri formate din mai mult de 2 elevi la un calculator se dovedesc a fi neproductive. Este bine ca la întocmirea grupurilor să se stabilească criterii clare de formare şi elevii să fie lăsaţi să se grupeze singuri, respectând criteriile cerute. Pentru grupurile omogene, sarcinile pot diferi în funcţie de scopul propus. Pentru grupurile eterogene sau create pe criterii afective, sarcinile vor fi aceleaşi la fiecare grup, dar profesorul va rezerva sarcini suplimentare elevilor mai buni din fiecare grup. Etapele pretinse de această metodă de învăţare sunt: repartizarea materialului (problemelor) fiecărui grup; munca independentă a grupurilor sub supravegherea profesorului; discutarea în plen a rezultatelor obţinute. Activitatea profesorului se concretizează în două etape. Prima este una proiectivă, în care se pregăteşte materialul de repartizat pe grupe şi materialul în plus pentru elevii buni, iar

a doua de îndrumare/supraveghere şi de animare a activităţii grupelor de lucru. Ajutorul dat grupelor de lucru trebuie să fie numai la cerere şi în aşa fel încât profesorul să se situeze pe poziţia de colaborator şi nu pe cea de autoritate care îşi impune părerile şi soluţia personală. Profesorul va interveni cu autoritate numai atunci când activitatea grupului se îndreaptă într-o direcţie greşită. Când unul sau mai multe grupuri descoperă o soluţie, propunerile lor vor fi discutate şi analizate succesiv sau în paralel. Scopul acestei discuţii este de a reliefa corectitudinea rezolvării, determinarea celei mai eficiente şi mai elegante soluţii şi de a descoperi eventualele erori. Importanţa dezbaterilor pentru dezvoltarea raţionamentului este foarte mare, iar rolul profesorului este acela de a incita şi coordona discuţiile în direcţia obţinerii concluziilor care se impun. Se impută, pe bună dreptate, acestei munci în grup o intensitate şi o productivitate scăzute. Diversificarea sarcinilor grupurilor şi împărţirea sarcinilor între membrii grupurilor atenuează această deficienţă. Dacă prin activitatea în grup se intenţionează dobândirea de noi cunoştinţe prin lucrul cu manualul, documentaţia sau produse soft, profesorul este obligat să organizeze dezbaterile finale care să stabilească dacă elevii şi-au însuşit corect noţiunile şi şi-au format deprinderi corecte. Este de asemenea greşit să se lucreze mereu cu grupuri constituite după aceleaşi criterii pentru că astfel sunt suprasolicitaţi elevii buni din grupurile eterogene, iar elevii slabi se bazează exclusiv pe aportul liderilor de grup, fie, în grupurile omogene, elevii slabi se complac în postura în care se află şi nu mai încearcă să scape de acest calificativ. Alte câteva probleme pot fi abordate sub un unghi diferit în acest context. Astfel, se pot pune întrebări mult mai individualizate (acestea nu ţin neapărat de conţinutul în sine al lecţiei). Ce întrebări se pot pune şi modul în care se pun poate fi mai important decât întrebarea în sine. Apoi, este mai simplă „contactarea" elevilor în timpul lecţiei şi chiar după ea. Susţinem, ca prioritate şi soluţie la anumite probleme locale de învăţământ aducerea unor specialişti care lucrează în lumea reală pentru a preda lecţii de sinteză, lecţii speciale etc.

2.9. Metoda lucrului cu manualul şi documentaţia

Manualele şcolare, purtătoare ale valenţelor formative prin deosebitul lor conţinut metodic şi didactic, reprezintă o limită impusă de programa şcolară din punctul de vedere al conţinutului informativ. În informatică, mai mult decât în alte domenii, manualul este supus perisabilităţii conţinuturilor prin frecvenţa cu care disciplina este receptivă la noutăţile domeniului. Realitatea didactică reliefează faptul că elevul foloseşte pentru învăţarea teoriei doar notiţele luate în clasă la predare şi, din considerente de comoditate sau de obişnuinţă, foarte puţin (sau deloc) manualele. Acestea sunt consultate în cel mai fericit caz doar pentru citirea enunţurilor problemelor. Atitudinea de reţinere sau de respingere faţă de manual are consecinţe negative atât asupra caracterului formativ, cât şi asupra celui informativ al învăţării. Capacitatea de raţionament a unui copil nu se formează numai după modele de raţionament oferite de profesor, ci şi prin eforturi proprii, prin activitatea proprie de căutare şi comparare cu alte scheme de raţionament. Valoarea acestei metode nu constă numai într-o însuşire temeinică a cunoştinţelor, ci şi în formarea unor deprinderi de activitate intelectuală. Mulţi elevi încheie ciclul liceal fără a avea deprinderi de lucru cu manualul şi documentaţia, ceea ce le creează serioase probleme de adaptare şi explică eşecurile din primul an de studenţie şi greutatea de adaptare la cerinţele studiului universitar. Metoda muncii cu manualul este un aspect al studiului individual şi se introduce ca metodă, treptat, sub directa îndrumare şi supraveghere a profesorului. Sunt discipline şi profesori care aplică în mod abuziv această metodă. Pe lângă efectele negative asupra învăţării, aceste abuzuri ascund şi alte aspecte care nu fac obiectul prezentei lucrări. Înainte de a aborda această metodă, profesorul trebuie să atragă atenţia elevului asupra aspectelor importante ale lecţiei, care trebuie urmărite în mod special, cerându-i să realizeze un rezumat cu principalele idei de reţinut. Rolul profesorului nu se limitează numai la a indica lecţia din manual sau documentaţia care trebuie studiată. În timpul studierii de către elevi a noului material, profesorul are un rol activ. El urmăreşte fiecare elev cum îşi urmăreşte conspectul, dă îndrumări cu voce scăzută elevilor care-l solicită, verifică planurile întocmite de aceştia, corectând acolo unde este cazul. Profesorul poate să descopere în acest fel anumite lacune

în cunoştinţele anterior dobândite ale elevilor şi să intervină ulterior pentru remedierea lor. El se ocupă deopotrivă de elevii slabi şi de cei buni cărora le dă sarcini suplimentare, reuşind să-şi facă o părere despre stilul de lucru şi ritmul fiecărui elev. După studierea individuală din manual sau documentaţie, urmează discuţii asupra celor însuşite de către elevi. Aceste discuţii au rolul de a preciza problemele esenţiale ale lecţiei, de a le sistematiza, da a înlătura posibilitatea unor omisiuni din partea elevilor sau chiar a însuşirii eronate a unor noţiuni. Profesorului i se cere o pregătire minuţioasă a materialului, pentru a fi în măsură să răspundă prompt la orice întrebare pusă de către elevi. Nu orice lecţie poate fi însuşită din manual. Metoda se aplică numai lecţiilor care au în manual o redactare sistematică şi accesibilă nivelurilor de vârstă şi de cunoştinţe ale elevilor. Metoda poate fi aplicată pentru studiul unor aplicaţii soft, limbaje procedurale (de exemplu HTML) sau în studiul comenzilor sistemelor de operare. Elevilor li se recomandă studiul temei stabilite pentru acomodarea cu noţiunile, apoi profesorul reia prezentarea cu sublinierea aspectelor esenţiale. Având o asemenea bază, profesorul se poate concentra asupra discursului său (ceea ce urmează este în strânsă legătură şi cu precedentele metode). Dacă este bine organizat, există următoarele avantaje:

Urmărirea atentă a audienţei: fiecărui „ascultător" îi poate fi sugerată ideea că este personajul principal, că el este vizat în primul rând.

Noi porţiuni de text pot fi uşor introduse suplimentar, prin referirea la „manual".

Se prezintă lucruri deja verificate. Nimic nu poate „merge rău", exceptând...îmbolnăvirea profesorului. Stresul fiecărui elev în parte poate fi micşorat, el ştiind că nu este „destinatarul" special. Există posibilitatea unui „feedback" rapid şi anumite principii de învăţare pot fi folosite imediat. Există posibilitatea pregătirii prealabile a materialului, cu durată determinată, inclusiv cea a expunerii. Posibilitatea de control asupra a ceea ce s-a transmis/recepţionat, cui, când, sub ce formă, precum şi a modului „de reacţie" este foarte mare.

Desigur că există şi dezavantaje. Nu insistăm, pentru că ideea este că fiecare avantaj de mai sus devine un dezavantaj dacă profesorul este un prost gestionar al metodelor şi timpului său. Oricum, se poate ajunge, din partea clasei, la pasivitate, stagnare, plictiseală, lipsă de individualizare etc.

2.10. Metoda jocurilor didactice

Jocurile didactice (şi nu numai) pe calculator au valenţele lor educative. Ca metodă de învăţare, jocurile didactice dau rezultate deosebite în special la clasele mici, dar marele pericol care planează asupra acestei metode îl constituie acele aplicaţii soft care au o încărcătură educativă redusă, dar prin atractivitate captează şi reţin atenţia elevului, uneori ore în şir, fără ca acesta să dobândească cunoştinţe sau deprinderi pe măsura efortului făcut. Un rol aparte se atribuie jocurilor manipulative, prin care elevul devine conştient de proprietăţile obiectului studiat, îşi formează deprinderi şi dexterităţi de utilizare a acestuia prin simularea pe calculator a utilajului sau dispozitivului respectiv. Aceste jocuri, numite uneori şi simulatoare, necesită în cele mai frecvente cazuri echipamente periferice suplimentare, unele specializate pe lângă cele clasice. Amintim în acest caz utilizarea unor căşti speciale pentru obţinerea efectului de realitate virtuală, echipamente care simulează condiţii de zbor etc. alte tipuri de jocuri, numite reprezentative, prin simbolizarea sau abstractizarea unor elemente reale, conduc la descoperirea unor reguli de lucru (sau joc) cu aceste elemente, dezvoltând în acest fel imaginaţia elevului. Ce altceva reprezintă un produs soft (de exemplu, un editor grafic sau de text) atunci când înveţi să-l utilizezi, decât un joc mult mai serios? Chiar dacă metoda nu este caracteristică studiului informaticii, la limita dintre jocul didactic şi învăţarea asistată de calculator se situează o bună parte dintre software-urile de învăţare, atât a informaticii, cât şi a altor discipline.

2.11. Instruirea programată şi învăţarea asistată de calculator

Instruirea programată poate fi aplicată cu mare succes în momentele în care obiectul primordial al predării îl constituie utilizarea unui mecanism real. În cadrul instruirii programate, esenţiale devin probele şi produsele demonstrative, pe care ar trebui să le descriem elevilor. Trebuie avut în vedere că numărul de ore afectat acestei instruiri programate să nu fie foarte mare. Acestea trebuie să includă un număr suficient de ore de verificare a cunoştinţelor acumulate, evitându-se însă monotonia şi instaurarea plictiselii (se recomandă utilizarea alternativă a altor metode). Trebuie evitată şi folosirea metodei un timp îndelungat, lucru care poate conduce în anumite situaţii la o izolare socială a elevului. O idee pentru contracararea acestor efecte ar fi creşterea numărului de ore sau organizarea activităţilor pe grupuri sau în echipă. Instruirea asistată de calculator este un concept diferit de instruirea programată doar prin modalitatea de utilizare. Există aceleaşi premise şi moduri de utilizare, cu excepţia faptului că un sistem de calcul devine principala interfaţă dintre un profesor şi un elev. Absolut toate noţiunile, conceptele, exerciţiile, problemele, evaluările, testările, prezentările legate de o anumită temă în cadrul unei lecţii (inclusiv estimarea îndeplinirii obiectivelor) sunt îndepliniri, dirijări, verificări cu ajutorul calculatorului (mediul soft corespunzător). Procesul de predare-învăţare şi verificare-evaluare funcţionează pe baza principiului cibernetic comandă-control-reglare (autoreglare). Instruirea programată, ca metodă didactică, presupune construirea unor programe care, prin fragmentarea materialului de studiat în secvenţe, realizează o adaptare a conţinuturilor la posibilităţile elevilor, la ritmul lor de învăţare, asigură o învăţare activă şi o informare operativă asupra rezultatelor învăţării, necesară atât elevului, pentru autocorectare, cât şi profesorului. În elaborarea programelor de învăţare se au în vedere următoarele operaţii:

- precizarea obiectivelor operaţionale în funcţie de conţinut şi posibilităţile elevilor;

structurarea logică a conţinutului după principiul paşilor mici şi al învăţării gradate;

- fracţionarea conţinutului în secvenţe de învăţare (unităţi didactice) inteligibile şi înlănţuite logic;
- fixarea după fiecare secvenţă a întrebărilor, exerciţiilor sau problemelor ce pot fi rezolvate pe baza secvenţei informaţionale însuşite;
stabilirea corectitudinii răspunsurilor sau soluţiilor elaborate; aceasta se poate realiza prin alegerea dintre mai multe răspunsuri posibile (trei, patru sau cinci). În situaţia în care nu s-a ales răspunsul corect, se poate recurge la întrebări suplimentare sau se elaborează un răspuns şi se compară cu cel corect.

Ca orice inovaţie, instruirea programată a trecut prin câteva faze contradictorii. La început s-a lovit de rezerva tenace a tradiţiei şi de dificultăţile materiale, apoi, după ce a câştigat teren în conştiinţa teoreticienilor şi practicienilor, s-au exagerat într-o oarecare măsură valenţele ei aplicative, creându-se iluzia descoperirii pietrei filozofale în domeniul pedagogic. În final, după o analiză lucidă, s-a admis că există părţi pozitive şi părţi negative. Criticile aduse instruirii programate sunt atât de ordin psihologic, cât şi de ordin pedagogic şi metodic. Psihologic, instruirii programate i se impută faptul că nu ţine seama de principiile psihologice ale învăţării, vizând învăţarea ca o simplă succesiune şi înmagazinare de fapte. De asemenea, se ştie că motivaţia învăţării nu poate fi analizată numai prin prisma reţinerii şi învăţării imediate, făcând abstracţie de interesul elevului faţă de conţinut. În plus, lucrând singur sau cu calculatorul, elevul se simte izolat. Pedagogic vorbind, fărâmiţarea conţinuturilor este în detrimentul formării unei viziuni globale, iar valoarea cunoaşterii imediate de către elev a rezultatului obţinut are valenţe contestabile. Metodic, decupajul analitico-sintetic al conţinuturilor îngustează elevului posibilitatea formării aptitudinilor de analiză şi sinteză. Aceste critici au determinat mutaţii serioase în concepţia de aplicare a metodei, dar practica didactică dovedeşte că atunci când se cunosc şi se evită cauzele care generează efecte negative, metoda produce rezultate bune. Tendinţele de îmbunătăţire a aplicării metodei se îndreaptă spre alternarea utilizării metodei cu celelalte metode clasice. Inserarea într-o lecţie programată a unor metode clasice schimbă determinarea muncii şcolare, repunându-l pe elev în directă dependenţă de activitatea profesorului şi dându-i acestuia posibilitatea să verifice gradul de însuşire a cunoştinţelor cunoscute în program. O altă tendinţă este aceea de a modifica modul de redactare al programului, în special prin mărirea volumului de informaţie din

unităţile logice şi prin separarea părţii de verificare, existând situaţii în care verificarea se va face după câteva ore sau chiar a doua zi. În plus, în program se pot insera secvenţe independente, care să necesite timp mai mare de gândire sau de lucru. Izolarea imputată învăţării programate poate fi contracarată prin alternarea cu munca în grup sau chiar prin învăţare programată în grup, situaţie în care grupul parcurge în colectiv un program special conceput în acest sens.

2.12. Metode de evaluare a rezultatelor şcolare

Evaluarea este activitatea prin care profesorul constată, în diferite momente ale procesului didactic, măsura în care rezultatele obţinute de către elevi sunt în concordanţă cu cerinţele programei, cu obiectivele pedagogice preconizate.

Evaluarea este o componentă esenţială a procesului de învăţământ, îndeplinind funcţii bine conturate:

- *funcţia de diagnoză (de constatare şi apreciere)*. Prin evaluare sunt furnizate informaţii despre nivelul rezultatelor obţinute în activitatea de predare-învăţare, comparativ cu obiectivele prevăzute;
- *funcţia de reglare.* Datele obţinute prin verificarea şi aprecierea rezultatelor şcolare stau la baza analizei cauzelor care au influenţat pozitiv sau negativ nivelul pregătirii şi a măsurilor de ameliorare a activităţii;
- *funcţia de predicţie şi orientare*. Pe baza rezultatelor controlului şi aprecierii, a interpretării acestora, se organizează activitatea didactică ulterioară, se prevăd direcţiile de desfăşurare care să asigure rezultatele scontate prin măsurile de ameliorare;
- *funcţia educativă.* Evaluarea, atunci când este corect realizată, are efecte pozitive asupra procesului de învăţare, orientează elevul spre conţinuturi semnificative, determină stilul de învăţare şi motivaţia, dezvoltă capacitatea de analiză, sinteză, autocontrol, contribuie la clarificarea şi aprofundarea cunoştinţelor;
- *funcţia de clasificare şi selecţie*. Potrivit rezultatelor evaluării, se realizează o ierarhizare a elevilor dintr-o clasă.

Orice proces de evaluare constă în a compara, a raporta un rezultat la o valoare prestabilită, la un etalon. În învăţământ, etalonul cu care se compară performanţele elevilor sunt obiectivele pedagogice.

Un prim demers în procesul de evaluare constă în verificarea rezultatelor. Prin verificare, profesorul determină elevul să exprime, să exteriorizeze, într-o formă sau alta, cunoştinţele pe care le deţine, capacitatea de a le aplica, de a opera cu acestea, modul de a gândi, de a raţiona.

Un alt demers îl reprezintă aprecierea, care este în strânsă legătură cu măsurarea. A măsura înseamnă a stabili numărul de caracteristici de o anumită natură pe care le întrunesc răspunsurile, performanţele elevilor verificaţi (numărul de răspunsuri corecte, numărul de greşeli, gravitatea acestora etc.). Numărul caracteristicilor şi valoarea acestora se raportează la obiectivele pedagogice şi, prin comparaţie, se stabileşte nota, se face aprecierea.

Principalele forme de evaluare întâlnite în practica didactică sunt:

- **evaluarea iniţială**, care conduce la formarea unei imagini despre bagajul de cunoştinţe cu care elevul porneşte la drum. Trebuie să ne asigurăm de ceea ce cunoaşte elevul înainte de a-l învăţa lucruri noi. Această formă de verificare creează şi o imagine asupra posibilităţilor de progres ale elevului, asupra capacităţii lui de învăţare, în funcţie de care se va stabili programul de instruire;
- **evaluarea formativă (continuă)** este forma de evaluare pe care profesorul o aplică pe întreaga durată a procesului de instruire, în cadrul lecţiilor şi la încheierea unui capitol. Această formă de verificare oferă permanent informaţii cu privire la eficienţa programului de instruire şi permite profesorului să ia cele mai potrivite măsuri de prevenire a insuccesului şcolar. Verificarea ritmică oferă, pe baza mecanismului de feed-back continuu, semnalele necesare atât elevului, cât şi profesorului, fiind un veritabil „metronom" al activităţii didactice;
- **evaluarea sumativă (cumulativă)** este forma tradiţională de evaluare realizată la sfârşitul unui semestru sau an şcolar şi cuprinde întreaga materie, conform programei şcolare, pe intervalul de timp la care se aplică verificarea. Rezultatele acestei forme de verificare nu reflectă întotdeauna adevăratul nivel de performanţă al elevilor, dar prin

faptul că determină o recapitulare şi o abordare globală a materiei parcurse, are efecte pozitive în direcţia dezvoltării capacităţii de cuprindere şi de sinteză a elevului.

Superioară prin caracterul ei predictiv, evaluarea formativă trebuie totuşi completată şi cu celelalte forme.

Rezultatele şcolare sunt obiectivate în cunoştinţele acumulate, în priceperi şi deprinderi, capacităţi intelectuale, trăsături de personalitate şi de conduită ale elevilor.

Metodele de verificare a randamentului şcolar presupun observarea modului în care învaţă elevul (logic, mecanic, creativ, ritmic, continuu, în salturi etc.) şi se realizează prin probe orale, scrise şi practice, teste de cunoştinţe şi deprinderi.

Verificarea orală, cea mai frecvent folosită, are anumite avantaje care o impun, prin faptul că favorizează dialogul, elevul putând să-şi argumenteze răspunsurile şi să participe la o confruntare de idei cu întreaga clasă, iar profesorul poate detecta cu uşurinţă erorile şi poate interveni şi corecta imediat. Verificarea orală are însă şi numeroase limite: întrebările nu au toate acelaşi grad de dificultate, unii elevi sunt emotivi şi se blochează, mai ales atunci când sunt ironizaţi de către profesor sau răspunsurile lor stârnesc ilaritate în clasă, timpul nu permite o verificare completă a conţinutului predat, iar comportamentul şi starea psihică a profesorului pot influenţa notarea.

Verificarea scrisă este utilizată sub forma unor lucrări de scurtă durată, lucrări „tip obiectiv", lucrări de una sau două ore, semestriale, care sunt înainte anunţate şi pregătite în clasă, respectiv lucrări de tip examen. Cercetările au dovedit că evaluarea formativă în formă scrisă, după fiecare capitol, şi combinată cu verificările orale este deosebit de eficientă şi stimulativă.

Probele scrise sunt preferate de elevi şi profesori pentru că asigură un grad mai mare de obiectivitate la notare, oferă elevilor mai emotivi sau celor care gândesc mai lent posibilitatea de a se exprima fără a fi influenţaţi de factori perturbatori, asigură evaluarea unui număr mare de elevi, întrebările au acelaşi grad de dificultate şi favorizează realizarea comparării rezultatelor. Dezavantajele metodei sunt marcate de faptul că profesorul nu poate interveni şi corecta pe loc erorile descoperite, el urmând să o facă în clasă, la discutarea lucrărilor. Elevii nu pot fi corectaţi dacă fac anumite confuzii sau când răspunsul nu este complet. Răspunsurile incomplete pot genera şi diferenţe de apreciere şi notare. Examinarea prin probe practice este caracteristică disciplinelor cu pronunţat

caracter aplicativ, cu atât mai mult informaticii. Ea se poate desfăşura în forme variate, de la realizarea unor programe simple, lucrându-se individual sau în grup, până la aplicaţii complexe, realizate într-un interval mai lung de timp. Sunt verificate şi evaluate cunoştinţele teoretice necesare realizării lucrării, cât şi deprinderile şi dexterităţile necesare executării ei.

O altă formă de verificare este evaluarea prin teste şi care se efectuează la începutul programului de instruire (iniţiale), pe parcursul acesteia (de progres) şi la sfârşitul programului (finale). Rezultatele acestor teste pot fi prelucrate statistic şi conduc la concluzii interesante în legătură cu eficienţa metodelor de predare-învăţare folosite.

Este necesară formarea la elevi a capacităţii de autoevaluare, prezentându-le criteriile de apreciere, ceea ce va mări încrederea elevului în propriile sale forţe şi va înlătura orice urmă de suspiciune.

Deşi imperfect, sistemul de evaluare permite o ierarhizare a elevilor în clase după criterii reale de competenţă, oferă informaţii edificatoare asupra nivelului de cunoştinţe al fiecărui elev, stimulează elevul la învăţătură.

Controlul cunoştinţelor dobândite de către elevi are o însemnătate deosebită, dă posibilitatea profesorului să dezvolte la elevi simţul răspunderii, să sesizeze la timp lipsurile, să aprecieze just munca elevilor. Controlul trebuie să fie sistematic, zilnic şi în mod echilibrat. La fiecare lecţie se verifică modul în care a fost înţeleasă şi asimilată lecţia nouă, iar cum lecţia are un caracter instructiv, trebuie verificat şi gradul în care cele expuse au fost reţinute.

Verificarea gradului de asimilare se face în diferite feluri:

- prin repetarea raţionamentelor de bază pe parcursul lecţiei cu sprijinul elevului (prin întrebări de control);
- prin rezolvarea de probleme;

Toate acestea ajută la verificarea rezultatelor muncii efectuate în clasă.

Verificarea lucrărilor scrise acasă se face:

- printr-o trecere (printre bănci) şi o examinare superficială, cantitativă;
- prin prezentarea rezolvării (ideea de rezolvare) de către un elev şi confirmarea de către ceilalţi elevi.

Este important ca verificarea temelor să se coreleze cu un set de întrebări, dinainte stabilite, care să verifice întreaga lecţie predată anterior.

Aceasta va permite elevilor să combine repetarea notiţelor predate cu formarea şi dezvoltarea deprinderilor de a corela noţiunile teoretice între ele şi de a le aplica în practică.

La verificarea problemelor este bine să se evidenţieze diferite variante de rezolvare a aceleiaşi probleme şi să se urmărească cele mai eficiente forme de rezolvare. De regulă se ascultă întâi expunerea coerentă a rezolvării, apoi se pun întrebările.

O altă formă de verificare este cea orală, cu toată clasa, când se pun întrebări, elevii sunt lăsaţi să se gândească, apoi este numit unul dintre ei care să răspundă. Ceilalţi sunt îndemnaţi să completeze răspunsul sau să corecteze greşelile. Această examinare, sumară de regulă, nu se notează, dar în situaţia în care un elev nu a învăţat deloc sau un altul a răspuns constant bine la mai multe întrebări, aceştia trebuie notaţi. La examinarea orală se pun întrebări care nu necesită desene, notări în caiete sau la tablă, calcule.

Examinarea cu scoaterea la tablă se face cu unul sau mai mulţi elevi. În timp ce elevii pregătesc răspunsurile, se poate lucra cu clasa sau verifica tema de acasă. Când elevii răspund, toată clasa este atentă şi pregătită să intervină. Profesorul poate să pună întrebări suplimentare sau ajutătoare, atât elevilor ascultaţi, cât şi celor din bănci. Prin întrebări se caută să se pună în evidenţă aspectele esenţiale ale lecţiei. Profesorul trebuie să-şi pregătească dinainte întrebările şi nu trebuie să transforme verificarea într-o scoatere cu sila la tablă şi punerea unui noian de întrebări care duc chiar până la sugerarea răspunsului.

Când elevul tace, profesorul nu trebuie să-i spună fraza sau ideea, ci să desemneze un alt elev. Intervenţia profesorului face ca să se creeze o ambiguitate cu privire la răspuns şi la cunoştinţele elevului.

După ce ascultă răspunsurile, profesorul pune note.

Lucrările de control scrise variază ca dimensiune astfel:

- cele scurte, de 10-15 minute, se dau în a doua parte a lecţiei şi urmăresc modul de asimilare a lecţiei noi sau a cunoştinţelor predate anterior, dar în corelaţie cu lecţia nouă;

- cele de 1-2 ore, se dau numai după ce au fost anunţate din timp şi pregătite, eventual, printr-o lecţie de recapitulare. Orice procedeu de verificare trebuie să îndeplinească anumite condiţii.

Verificarea trebuie:

- să aibă un scop precis, care, chiar dacă nu este transparent pentru elev, el trebuie să fie foarte clar pentru profesor;
- să dezvolte deprinderea elevului de a raţiona rapid şi de a da răspunsuri corecte, precise şi scurte, dar complete;
- să dezvolte la elevi grija pentru formulări exacte şi exprimări corecte din punct de vedere ştiinţific şi gramatical;
- să permită elevilor să aprecieze răspunsurile;
- să fie operativi (să nu ia mult timp).

3. PROIECTAREA ACTIVITĂŢII DIDACTICE

În practica didactică s-a stabilit ca profesorul să prezinte o planificare anuală (calendaristică). Planificarea calendaristică trebuie să conţină eşalonarea conţinutului disciplinei pentru care se realizează. În planificarea calendaristică se vor face referiri la materialul didactic şi la lucrările practice care vor fi efectuate. Rubricaţia planificării calendaristice depinde de gradul de detaliu la care se doreşte să se realizeze aceasta. Temele specificate în planificare sunt concretizate în lecţii, pentru care profesorul trebuie să întocmească un proiect de tehnologie didactică la nivel de detaliu.

Un prim demers în elaborarea unui proiect de lecţie constă în analiza conţinuturilor învăţării delimitat de programă şi a obiectivelor corespunzătoare.

Profesorul va analiza conţinuturile învăţării corelat cu obiectivele propuse de programă pentru a stabili abilităţile, calităţile proceselor psihice, ale motricităţii şi afectivităţii care se pot dezvolta prin însuşirea conţinutului respectiv. Şi nu în ultimul rând, pentru a stabili sarcinile didactice, adică acţiunile cele mai potrivite şi succesiunea acestora pentru ca elevii să-şi dezvolte capacităţile pe care le prilejuieşte însuşirea acestor cunoştinţe, să se atingă performanţele prescrise prin obiective.

Deoarece lecţia este un act de creaţie, ce nu se poate încadra în şabloane, acestea furnizează doar sugestii pentru întocmirea de scenarii didactice. Lecţia se caracterizează prin mobilitatea structurii şi prin flexibilitate.

3.1. Lecţie pentru formare şi consolidare de deprinderi si priceperi

LICEUL: COLEGIUL NAŢIONAL „VLAICU VODĂ“ CURTEA DE ARGEŞ

DISCIPLINA : Informatică

CLASA a XI-a

Unitatea de învăţare : Parcurgerile grafurilor neorientate

Tema : Parcurgerea grafului în lăţime şi în adâncime .

Tipul lecţiei : formarea şi consolidarea de deprinderi si priceperi

Locul de desfăşurare : laboratorul de informatică

Nivelul iniţial al clasei :

- elevii şi-au însuşit toate noţiunile legate de modul de reprezentare a grafurilor neorientate .
- elevii utilizează corect noţiunile anterioare şi modurile de reprezentare a grafurilor neorientate .

Obiectiv cadru: realizarea de aplicaţii utilizând parcurgerile grafurilor neorientate

Obiective de referinţă :

- să aplice parcurgerile prezentate
- să utilizeze în algoritmi parcurgerea cea mai avantajoasă din punct de vedere al optimului

Obiective educaţionale :

- obiective cognitive :
  - să utilizeze corect parcurgerile
  - să recunoască în aplicaţii ce parcurgere s-a utilizat
  - să compare variantele de parcurgere
  - să analizeze modul diferit de funcţionare a aplicaţiilor în funcţie de parcurgerea utilizată

- obiective afective :
  - să argumenteze corect alegerea unei parcurgeri, utilizate într-o aplicaţie
  - să aprecieze corect soluţiile la problemele oferite de alţi colegi
  - să se autoevalueze corect în raport cu clasa
- obiective psihomotorii :
  - sa utilizeze corect noţiunile teoretice însuşite
  - sa deosebească modalităţile de parcurgere
  - să implementeze algoritmi care să utilizeze parcurgeri ale grafurilor într-un limbaj de programare

Obiective operaţionale :

1) să utilizeze corect parcurgerile
2) să-şi însuşească corect modul de funcţionare al parcurgerilor
3) să analizeze corect fiecare problemă pentru a alege cea mai bună modalitate de parcurgerea grafurilor neorientate

Strategii didactice :

- Principii didactice :
- Principiul participării şi învăţării active
- Principiul asigurării progresului gradat al performantei
- Principiul conexiunii inverse
- Metode de învăţământ :
- Metode de comunicare orală: expunere, conversaţie, problematizare
- Metode de acţiune : exerciţiul, învăţare prin descoperire
- Procedee de instruire :
  - Explicaţia in etapa de comunicare
  - Învăţarea prin descoperire, prin rezolvare de probleme
  - Problematizarea prin creare situaţiilor problema
  - Conversaţia de consolidare în etapa de fixare a cunoştinţelor

- Forme de organizare : frontală si individuală
- Forme de dirijare a învăţării : dirijată de profesor sau independentă
- Resurse materiale :
  - George Daniel Mateescu – Informatica – Manual pentru clasa a XI-a Editura Niculescu
  - Cornelia Ivasc, Mona Pruna – Bazele Informaticii – Proiect de manual – Editura Petrion
- Metode de evaluare :
  - Evaluare iniţială : test grilă
  - Set de aplicaţii
- Desfăşurarea lecţiei :
  - Moment organizatoric
    - Pregătirea lecţiei :
      - Întocmirea proiectului didactic
      - Pregătirea setului de întrebări
      - Pregătirea setului de aplicaţii
      - Pregătirea temei
    - Organizarea si pregătirea clasei :

Verificarea frecvenţei

- Captarea atenţiei clasei :

Anunţarea subiectului pentru tema respective

Anunţarea obiectivelor urmărite

Anunţarea modului de desfăşurare a activităţii

- Reactualizarea cunoştinţelor

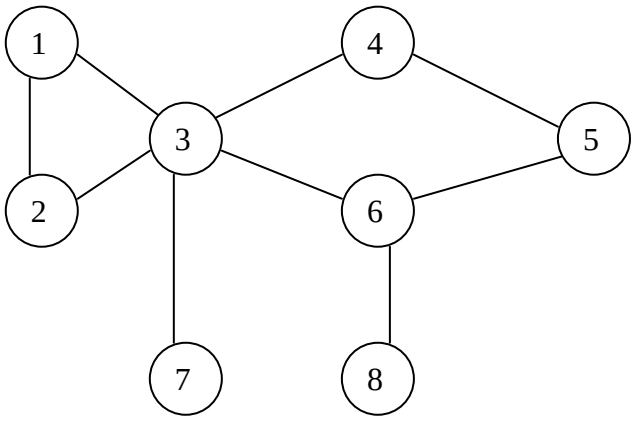
Se realizează un set de întrebări pentru reactualizarea cunoştinţelor teoretice ca în tabelul de mai jos :

Nr. crt	Întrebare	Răspunsul aşteptat
1.	Care dintre următoarele afirmaţii fac referire la matricea de adiacenţă a unui graf neorientat : a) Este simetrică faţă de diagonala secundară b) Pe o linie sau pe o coloana toate elementele pot fi egale cu 1 . c) Este simetrică faţă de diagonala principală d) Suma elementelor de pe linia i este diferită de suma elementelor de pe coloana i	c)
2.	Suma elementelor de deasupra diagonalei principale dintr-o matrice de adiacenţă este egala cu : a) Dublul numărului de muchii b) Numărul de muchii c) Numărul nodurilor grafului d) Suma gradelor nodurilor	b)
3.	Care din următoarele afirmaţii este adevărată : a) Matricea de adiacenţă nu este pătratică b) Numărul de componente al vectorului de muchii este mai mare strict decât $\frac{n(n-1)}{2}$ c) Gradul maxim al unui nod dintr-un graf neorientat cu n vârfuri este n d) Suma elementelor dintr-o matrice de adiacenţă este egală cu dublul numărului de muchii .	d)

4.	Care dintre următoarele afirmaţii este falsă : a) Dacă toate nodurile unui graf neorientat cu n vârfuri au gradul n-1 atunci graful este complet b) Orice graf neorientat are grafuri parţiale şi subgrafuri c) Diagonala secundară a matricei de adiacenţă poate avea toate elementele nule d) Numărul maxim de componente ale vectorului de muchii este $\frac{n(n+1)}{2}$	d)

- Parcurgeri de grafuri neorientate . Exemple.

Se prezintă un graf sub forma grafică ca în figura alăturată :



şi se pun următoarele probleme

:

1. Care este parcurgerea grafului în lăţime începând cu nodul 2 ?

Răspuns aşteptat : 2,1,3,4,6,7,5,8

2. Care este parcurgerea grafului în lăţime începând cu nodul 3 ?

Răspuns aşteptat : 3,1,2,4,6,7,5,8

3. Care este parcurgerea grafului în adâncime începând cu nodul 7 ?

Răspuns aşteptat : 7,3,1,2,4,5,6,8

4. Care este parcurgerea grafului în adâncime începând cu nodul 5 ?

Răspuns aşteptat : 5,4,3,1,2,7,6,8

- Obţinerea performanţei .

Set de probleme în care se utilizează parcurgerile grafurilor :

1. Se dă un graf neorientat prin matricea de adiacenţă citită dintr-un fişier text şi nouă noduri x respectiv y. Să se verifice dacă între cele două noduri există lanţ care să le unească .
2. Se dă un graf neorientat prin matricea de adiacenţă citită dintr-un fişier text . Să se determine lungimile lanţurilor minime de la un nod x, citit de la tastatură, la celelalte noduri ale grafului .

Rezolvările aşteptate :

1.

```
#include<fstream.h>
int a[30][30], viz[30], n,x,y,i,j,v,p,u, c[30];
void citire(int a[30][30],int &n, int &x, int &y)
{int  i,j;
   ifstream f (”graf.in”);
   f>>n;
```

```
  for (i=1; i<=n; i++)
    for (i=1; i<=n; i++)
             f>>a[i][j]);
           f>>x>>y;
  f.close();
}
void main()
{
  citire(a,n,x,y);
  c[1]=x;
  p=1;
  u=1;
  viz[x]=1;
  while p<=u do
   {
       v:=c[p];
      for (i=1; i<=n; i++)
        if (a[i][v]==1 && viz[i]==0)
                     {
                         u++;
                          c[u]=i;
                          viz[i]=1;
                       }
      p++;   }
  if (viz[y]==1)   cout<<"exista lant");
  else    cout<<"nu exista lant"); }
```

2.

```
#include<fstream.h>
int a[30][30], viz[30], n,x,y,i,j,v,p,u, c[30];
void citire(int a[30][30],int &n)
{int  i,j;
   ifstream f ("graf.in");
```

```
   f>>n;
  for (i=1; i<=n; i++)
    for (i=1; i<=n; i++)
            f>>a[i][j]);
            f.close();}
void main()
{  citire(a,n);
cout<<"dati nodul";
  cin>>x;
  c[1]=x;
  for (i=1 ; i<= n ; i++)
    viz[i]=-1
    p=1;
  u=1;
  viz[x]=0;
  while p<=u do
   {       v=c[p];
      for (i=1; i<=n ; i++)
        if (a[i][v]==1 && viz[i]==0)
                    {
                        u++;
                         c[u]=i;
                         viz[i]=viz[v]+1;
                      }
      p++;
   }
  for( i=1 ; i<=n ; i++)
    if (viz[i]=-1)   cout<< "nu exista lant de la "<<x<<"
la"<<i);
    else cout<<" lungimea minima de la "<<x<<" la"<<
i<<",viz[i])";
} }
```

3.2. Lecţie pentru recapitulare şi sistematizare

LICEUL: COLEGIUL NAŢIONAL „VLAICU VODĂ“ CURTEA DE ARGEŞ

DISCIPLINA : Informatică

CLASA a XI-a

Unitatea de învăţare : Conexitate în grafuri neorientate

Tema : Noţiunile preliminarii : lanţ, ciclu, componenta conexă, graf conex

Tipul lecţiei : recapitulare şi sistematizarea cunoştinţelor

Locul de desfăşurare : laboratorul de informatică

Nivelul iniţial al clasei :

- elevii şi-au însuşit toate noţiunile legate de lanţ, ciclu şi componentă conexă
- elevii utilizează corect noţiunile definite anterior cu privire la lanţ elementar şi neelementar, ciclul elementar şi neelementar şi a componentelor conexe

Obiectiv cadru : realizarea de aplicaţii utilizând noţiunile legate de conexitate

Obiective de referinţa :

- să aplice corect noţiunile învăţate
- să utilizeze în algoritmi parcurgerile grafurilor pentru studiul conexităţii

Obiective educaţionale :

- obiective cognitive :
  - să definească corect noţiunile : lanţ, ciclu, componentă conexă şi graf conex
  - să recunoască în aplicaţii utilizarea lanţurilor sau a ciclurilor
  - să compare variantele de obţinere a componentelor conexe în funcţie de parcurgerea utilizată
  - să analizeze modul diferit de formare a componentelor conexe în funcţie de parcurgeri
- obiective afective :

- să argumenteze corect alegerea unei parcurgeri pentru studiul conexităţii grafului
- să aprecieze corect soluţiile la problemele oferite de alţi colegi
- să se autoevalueze corect în raport cu clasa

➢ obiective psihomotorii :

- să-şi formeze deprinderi de lucru specifice temei de lucru
- să-şi dezvolte gândirea logică, capacitatea de generalizare şi problematizare
- să implementeze algoritmi care să utilizeze componentele conexe ale grafurilor într-un limbaj de programare

Obiective operaţionale :

1) să utilizeze corect noţiunile de lanţ şi ciclu
2) să-şi însuşească corect modul de obţinere a componentelor conexe
3) să analizeze corect fiecare problemă pentru a alege cea mai bună modalitate de a verifica dacă un graf este conex sau nu

Strategii didactice :

❑ Principii didactice :

- Principiul participării şi învăţării active
- Principiul asigurării progresului gradat al performanţei
- Principiul conexiunii inverse

❑ Metode de învăţământ :

- conversaţia
- explicaţia
- exerciţiul
- problematizarea
- algoritmizarea

❑ Procedee de instruire :

- Conversaţia de consolidare
- Exerciţii de consolidare
- Problematizarea prin creare situaţiilor problemă

❑ Forme de organizare : frontală, individuală şi pe grupe

- Forme de dirijare a învăţării : dirijată de profesor sau prin materiale didactice respectiv independentă
- Resurse materiale :
  - Fişe de lucru
  - Set de aplicaţii
  - calculatorul
- Metode de evaluare :
  - Evaluare sumativă
  - Evaluare continuă pe parcursul lecţiei (fişa de lucru şi calculatorul)
  - Evaluare formativă
- Desfăşurarea lecţiei :
  - Moment organizatoric
    - Pregătirea lecţiei :
      - Întocmirea proiectului didactic
      - Pregătirea setului de întrebări
      - Pregătirea setului de aplicaţii
      - Pregătirea temei
    - Organizarea si pregătirea clasei :

Verificarea frecvenţei

Verificarea cantitativă si calitativă a temei

Verificarea existenţei şi operaţionalităţii resurselor materiale

- Captarea atenţiei clasei :

Anunţarea subiectului pentru tema respective

Anunţarea obiectivelor urmărite

Anunţarea modului de desfăşurare a activităţii

- Reactualizarea cunoştinţelor

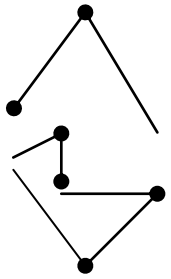
Se realizează un set de întrebări pentru reactualizarea cunoştinţelor teoretice ca în tabelul de mai jos :

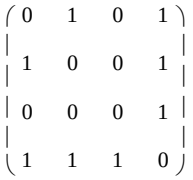
Nr. crt	Întrebare	Răspunsul aşteptat
1.	Care este definiţia lanţului ?	Lanţul este o succesiune de noduri cu proprietatea ca oricare două noduri consecutive sunt adiacente .
2.	Câte tipuri de lanţuri există ?	Lanţurile sunt de două tipuri elementare şi neelementare. Dacă nodurile unui lanţ sunt distincte două câte două atunci el este elementar iar în caz contrar neelementar .
3.	Care este definiţia ciclului ?	Ciclul este un lanţ în care primul nod coincide cu ultimul, iar oricare două muchii sunt distincte între ele .
4.	Câte tipuri de cicluri există ?	Ciclurile sunt de două feluri elementare şi neelementare. Dacă nodurile unui ciclul, mai puţin primul şi ultimul, sunt distincte două câte două atunci el este elementar în caz contrar el fiind neelementar .
5.	Ce este un graf conex ?	Un graf este conex dacă între oricare două noduri ale grafului, noduri diferite , există lanţ care să le unească .
6.	Ce este o componentă conexă ?	Componenta conexă a unui graf este un subgraf conex al grafului .
7.	Câte componente are un graf conex ?	Un graf conex are o singura componentă conexă care este chiar graful însuşi .
8.	Care este numărul minim de muchii ce trebuie adăugate astfel încât graful să devină conex ?	Într-un graf numărul minim de muchii ce trebuie adăugat pentru a deveni conex este egal cu numărul componentelor conexe minus unu .

- Fişă de lucru :

Este prezentată sub forma tabelului următor :

Fisa 1. - Test grilă

Nr. Crt	Întrebare	Răspuns
1.	Se consideră un graf neorientat cu 7 noduri si 3 muchii . Numărul de componente conexe din care poate fi format graful este : Exact 4 4 sau 5 3 sau 4 cel puţin 5	b)
2.	Orice graf neorientat cu n noduri şi n-1 muchii a) este aciclic si neconex b) este conex dacă şi numai dacă este aciclic c) este un arbore d) este conex si conţine un ciclu	b)
3.	Dacă într-un graf neorientat conex cu n noduri, lista de adiacenţă a fiecărui nod este formată din exact două elemente, atunci în graful respectiv există: a) cel puţin n/2 cicluri b) exact n/2 cicluri c) exact un ciclu d) cel puţin doua cicluri	c)
4.	Numărul minim de muchii ce se pot alege pentru a fi eliminate din graful neorientat alăturat astfel încât acesta să devină neconex este :  a) 2 b) 4 c) 1	a)

	d) 3	
5.	Se consideră graful neorientat dat prin matricea de adiacenţă alăturată . În graf nodurile sunt numerotate de la 1 la 4, corespunzător liniilor tabloului. Să se determine lungimea minimă a unui lanţ ce uneşte nodurile 1 si 3 .  a) 1 b) 4 c) 3 d) 2	d)

Fisa 2. – Probleme

1. Se consideră un graf neorientat prin matricea de adiacenţă. Se citeşte un şir de p noduri. Să se verifice dacă şirul formează un lanţ, iar în caz afirmativ să se specifice natura .

2. Se consideră un graf neorientat prin matricea de adiacenţă. Se citeşte un sir de p noduri. Să se verifice dacă şirul formează un ciclu, iar în caz afirmativ să se specifice natura .

3. Se consideră un graf neorientat prin matricea de adiacenţă. Să se specifice un număr minim de muchii ce ar trebui adăugate astfel încât graful sa devină conex .

Rezolvări aşteptate :

1.

```
#include<fstream.h>
```

```
int a[30][30], n,p,x[30];
void citire(int a[30][30],int &n, int x[30], int &p)
{int  i,j;
   ifstream f (”graf.in”);
   f>>n;
  for (i=1; i<=n; i++)
    for (i=1; i<=n; i++)
            f>>a[i][j]);
           f>>p;
  for (i=1; i<=p; i++)
f>>x[i];
  f.close();
}
int lant(int x[30];int p)
{
int i;
     for (i=1; i<p; i++)
    if (a[x[i]][x[i+1]]==0) return 0;
return 1;
}
int  natura(int x[30], int p)
{
int  i,j;
   for (i=1; i<p ;  i++)
     for (j=i+1 ; j<=p ; j++)
      if( x[i]==x[j]) return 0;
return 1;
}
void main()
{
  citire(a,n,x,p);
  if (lant(x,p))
```

```
                if (natura(x,p))
                                    cout<<”este lant elementar”;
                else cout<<” este lant neelementar”;
  else cout<< ”nu este lant”;
}
```

2.

```
#include<fstream.h>
int a[30][30], n,p,x[30];
void citire(int a[30][30],int &n, int x[30], int &p)
{int  i,j;
   ifstream f (”graf.in”);
   f>>n;
  for (i=1; i<=n; i++)
    for (i=1; i<=n; i++)
             f>>a[i][j]);
           f>>p;
  for (i=1; i<=p; i++)
f>>x[i];
  f.close();
}
int  ciclu(int x[30], int p)
{
int i;
    if (x[1]!=x[p])  return 0;
  for (i=1 ; i< p ; i++)
    if ( a[x[i]][x[i+1]]==0 ) return 0;
  return 1;
}
int  natura(int x[30], int p)
{
int  i,j;
```

```
for (i=1 ; i< p-1 ; i++)
  for ( j=i+1 ; j< p ; j++)
      if ( x[i]==x[j])  return 0;
return 1;
}
void main()
{
citire(a,n,x,p);
  if ( ciclu(x,p))
              if (natura(x,p))
                                  cout<<      ”este      ciclu
elementar”;
              else cout<<”este ciclu neelementar”;
  else  cout<<” nu este ciclu ”;}
```

3.

```
#include<fstream.h>
int a[30][30], n,p,y[30],i,j,c[30],nc,v,m,u,viz[30];
void citire(int a[30][30],int &n,  int &m)
{int  i,j;
   ifstream f (”graf.in”);
   f>>n>>m;
  for (i=1; i<=n; i++)
    for (i=1; i<=n; i++)
            f>>a[i][j]);
       f.close();
}
int  nod(int viz[30])
{
int  i;
  for( i=1 ; i<=n ; i++ )
    if (viz[i]==0)  return i;
        return 0;}
```

```
void main()
{  citire(a,n,m);
  x=nod(viz);
  while( x>0)
    {
      nc++;
      c[1]=x;
      y[nc]=x;
      viz[x]=1;
      p=1;
      u=1;
      while( p<=u)
        {
          v=c[p];
          for( i=1; i<=n ; i++ )
          if (a[v][i]==1 && viz[i]==0)
                  {
                      u++;
                      c[u]=i;
                      viz[i]=1;
                   }
          p++;
        }
        x=nod(viz);
      }
  cout<< " nr. minim de muchii ce trebuie adaugate ca sa
devina conex ="<<nc-1;
  cout<< "exemplu de muchii ");
  for (i=1; i<nc ; i++)
    cout<<"("<<y[i]<<","<<,y[i+1]<<")";
}
```

Fisa 3. – Probleme pentru obţinerea performanţei

Obţinerea performanţei se realizează prin utilizarea fişei cu numărul 3, propusă pentru elevii capabili de performanţă care o pot realiza în clasă şi constituind temă pentru acasă celorlalţi elevi .

1. Se dă un graf neorientat prin matricea de adiacenţă. Să se verifice dacă trei noduri x, y şi z fac parte din aceeaşi componentă conexă.

2. Se dă un graf neorientat prin matricea de adiacenţă . Să se determine lungimile drumurilor minime de la un nod dat x la celelalte noduri.

3.3. Fişe de lucru pentru pregătirea concursurilor şcolare

3.3.1. Probleme rezolvate

1. Scara (Olimpiada de Informatică, faza judeţeană 2005)

Domnul G are de urcat o scară cu n trepte. În mod normal, la fiecare pas pe care îl face, el urcă o treaptă. Pe k dintre aceste trepte se află câte o sticlă cu un număr oarecare de decilitri de apă, fie acesta x. Dacă bea toată apa dintr-o astfel de sticlă, forţa şi mobilitatea lui G cresc, astfel încât, la următorul pas el poate urca până la x trepte, după care, dacă nu bea din nou ceva, revine la "normal". Sticlele cu apă nu costă nimic. Cantitatea de apă conţinută de aceste sticle poate să difere de la o treaptă la alta.

Pe j trepte se află câte o sticlă cu băutura energizantă. Şi pentru aceste sticle, cantitatea de băutură energizantă poate să difere de la o treaptă la alta. Să presupunem că într-una dintre aceste sticle avem y decilitri de băutură energizantă. Dacă bea q ($q \leq y$) decilitri dintr-o astfel de sticlă, la următorul pas G poate urca până la 2q trepte, după care şi în acest caz, dacă nu bea din nou ceva, el revine la "normal". Însă băutura energizantă costă: pentru o cantitate de q decilitri consumaţi, G trebuie să plătească q lei grei.

Pot exista trepte pe care nu se află nici un pahar, dar şi trepte pe care se află atât o sticlă cu apă cât şi una cu băutură energizantă. În astfel de situaţii, nu are rost ca G să bea

ambele băuturi deoarece efectul lor nu se cumulează; el poate alege să bea una dintre cele două băuturi sau poate să nu bea nimic.

Cerinţă

Determinaţi p, numărul minim de paşi pe care trebuie să îi facă G pentru a urca scara, precum şi suma minimă pe care trebuie să o cheltuiască G pentru a urca scara în p paşi.

Date de intrare

Fişierul text de intrare scara.in conţine:

- pe prima linie un număr natural n, reprezentând numărul total de trepte;
- pe cea de a doua linie un număr natural k, reprezentând numărul de trepte pe care se află sticle cu apă;
- pe fiecare dintre următoarele k linii câte două numere naturale separate printr-un spaţiu, reprezentând numărul de ordine al treptei pe care se află o sticlă cu apă şi respectiv cantitatea de apă din acea sticlă exprimată în decilitri;
- pe următoarea linie un număr natural j, reprezentând numărul de trepte pe care se află sticle cu băutură energizantă;
- pe fiecare dintre următoarele j linii câte două numere naturale separate printr-un spaţiu, reprezentând numărul de ordine al treptei pe care se află o sticlă cu băutură energizantă şi respectiv cantitatea de băutură energizantă din acea sticlă exprimată în decilitri.

Date de ieşire

Fişierul text de ieşire scara.out va conţine o singură linie pe care vor fi scrise două numere naturale p c separate printr-un spaţiu, p reprezentând numărul minim de paşi, iar c suma minimă cheltuită.

Restricţii

$n \leq 120$

$0 \leq k \leq n$

$0 \leq j \leq n$

Cantitatea de apă aflată în oricare sticlă este $1 \leq x \leq 100$

Cantitatea de băutură energizantă aflată în oricare sticlă este $1 \leq y \leq 100$

Exemple

scara.in	scara.out	scara.in	scara.out
6	3 2	6	4 1
1		1	
1 2		1 2	
2		2	
4 1		4 1	
1 2		1 1	

Descrierea soluţiei

Se construieşte un graf orientat cu n noduri (care corespunzător celor n trepte) la arcele căruia se ataşează costuri urmărind ca:

- între două noduri costul să fie cu atât mai mic cu cât nodurile sunt "mai depărtate"
- costul pentru salturi efectuate după ce se bea băutura energizantă să fie mai mare decât costul salturilor făcute după ce se bea apă între aceleaşi noduri, dar acest cost să nu depaşească costul saltului între două noduri mai apropiate.

În calculul costurilor între doua noduri am folosit formulele:

cost[i][i+1] = 999000

cost[i][j]= 1000000 – (j-i) pentru salturi când se bea apă j>i

cost[i][j]= 1000000 – (j-i) + q pentru salturi când se bea energizantă j>i, $1 \le q \le y$

Pe acest graf se aplică apoi un algoritm de aflare a drumului minim de sursă unică

Rezolvare

```
#include <fstream.h>
#include <values.h>
const nmax=121;
unsigned long cost[nmax+1][nmax+1];
unsigned long c[nmax+1],pd[nmax+1],d[nmax+1],min;
int eng[nmax+1];
void main()
{ifstream fi("scara.in");
 ofstream fo("scara.out");
```

```
 int n,k,j,i,x,cx,i1,nod1,nod2,y,p,v;
 fi>>n;
 for (i=0;i<n+1;i++)
  for (j=0;j<n+1;j++)
         cost[i][j]=MAXLONG;
//costul sariturii pe prima treapta
 cost[0][1]=999000;
//costurile cand merge din 1 in 1
 for (i=1;i<=n;i++)
  cost[i][i+1]=999000;
 fi>>k;
 for (i=0;i<k;i++)
  { fi>>p>>x;
    if (! fi.good() || x>100) cout<<"apa incorect\n";
          nod1=p;
          for (nod2=nod1+2,i1=2;i1<=x;i1++,nod2=nod2+1)
           if (nod2<=n)
            cost[nod1][nod2]=1000000-100*(nod2-nod1);
  }
//costurile cand bea energizanta
 fi>>j;
 for (i=0;i<j;i++)
  { fi>>p>>y;
    if (! fi.good() || y>100) cout<<"baut incorect\n";
    eng[p]=y;
  }
 for (i=1;i<=n;i++)
  if (eng[i]!=0)
  {
          nod1=i;
          for (nod2=nod1+1,i1=1;i1<=eng[i];i1++,nod2=nod2+2)
```

```
        {if (nod2<=n)
              if (cost[nod1][nod2]>1000000-100*(nod2-nod1)+i1)
                    cost[nod1][nod2]=1000000-100*(nod2-nod1)+i1;
         if (nod2+1<=n)
              if (cost[nod1][nod2+1]>1000000-100*(nod2-nod1+1)+i1)
                    cost[nod1][nod2+1]=1000000-100*(nod2-nod1+1)+i1;
        }
  }
for (i=1;i<n+1;i++)
{  c[i]=1;
      d[i]=cost[0][i];
}
pd[1]=0;
for (j=0;j<n-1;j++)
  for (i=1;i<n+1;i++)
      if (c[i]!=0)
       if (min>d[i])
        {min=d[i];
              v=i;
         }
       c[v]=0;
       for (i=1;i<n+1;i++)
        if (c[i]!=0)
              if (d[i]>d[v]+cost[v][i])
               {
                    d[i]=d[v]+cost[v][i];
                    pd[i]=v;
               }
 }
 i=n;
 int ct=0,cst=0;
```

```
 while (i!=0)
        {j=pd[i];
         ct++;
         if (cost[j][i]%100!=0)
                cst = cst+cost[j][i]%100;
         i=j;
        }
 fo<<ct<<' '<<cst<<endl;
 fo.close();
}
```

2. Cezar (Olimpiada de Informatică, faza judeţeană 2007)

În Roma antică există **n** aşezări senatoriale distincte, câte una pentru fiecare dintre cei **n** senatori ai Republicii. Aşezările senatoriale sunt numerotate de la **1** la **n**, între oricare două aşezări existând legături directe sau indirecte. O legătură este directă dacă ea nu mai trece prin alte aşezări senatoriale intermediare. Edilii au pavat unele dintre legăturile directe dintre două aşezări (numind o astfel de legătură pavată ”stradă“), astfel încât între oricare două aşezări senatoriale să existe o singură succesiune de străzi prin care se poate ajunge de la o aşezare senatorială la cealaltă.

Toţi senatorii trebuie să participe la şedinţele Senatului. In acest scop, ei se deplasează cu lectica. Orice senator care se deplasează pe o stradă plăteşte **1** ban pentru că a fost transportat cu lectica pe acea stradă.

La alegerea sa ca prim consul, Cezar a promis că va dota Roma cu o lectică gratuită care să circule pe un număr de **k** străzi ale Romei astfel încât orice senator care va circula pe străzile respective, să poată folosi lectica gratuită fără a plăti. Străzile pe care se deplasează lectica gratuită trebuie să fie legate între ele (zborul, metroul sau teleportarea nefiind posibile la acea vreme).

În plus, Cezar a promis să stabilească sediul sălii de şedinţe a Senatului într-una dintre aşezările senatoriale aflate pe traseul lecticii gratuite. Problema este de a alege cele **k** străzi şi amplasarea sediului sălii de şedinţe a Senatului astfel încât, prin folosirea transportului gratuit, senatorii, în drumul lor spre sala de şedinţe, să facă economii cât mai însemnate. În calculul costului total de transport, pentru toţi senatorii, Cezar a

considerat că fiecare senator va călători exact o dată de la aşezarea sa până la sala de şedinţe a Senatului.

Cerinţă

Scrieţi un program care determină costul minim care se poate obţine prin alegerea adecvată a celor **k** străzi pe care va circula lectica gratuită şi a locului de amplasare a sălii de şedinţă a Senatului.

Date de intrare

Fişierul **cezar.in** conţine

- pe prima linie două valori **n k** separate printr-un saţiu reprezentând numărul total de senatori şi numărul de strazi pe care circulă lectica gratuită
- pe următorele **n-1** linii se află câte două valori **i j** separate printr-un spaţiu, reprezentând numerele de ordine a două aşezări senatoriale între care există stradă.

Date de ieşire

Pe prima linie a fişierului **cezar.out** se va scrie costul total minim al transportării tuturor senatorilor pentru o alegere optimă a celor **k** străzi pe care va circula lectica gratuită şi a locului unde va fi amplasată sala de şedinţe a Senatului.

Restricţii

– $1<n\leq10000$, $0<k<n$

– $1\leq i,j\leq n$, $i\neq j$

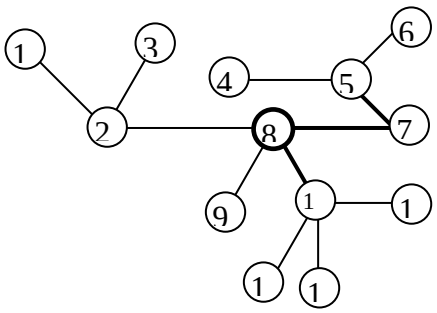
– Oricare două perechi de valori de pe liniile **2,3,...,n** din fişierul de intrare reprezintă două străzi distincte.

– Perechile din fişierul de intrare sunt date astfel încât respectă condiţiile din problemă.

– Pentru 25% din teste $n\leq30$, pentru 25% din teste $30<n\leq1000$, pentru 25% din teste $1000<n\leq3000$, pentru 10% din teste $3000<n\leq5000$, pentru 10% din teste $5000<n\leq10000$.

Exemplu

cezar.in	cezar.out	Explicaţie
13 3 1 2 2 3 2 8 7 8 7 5 5 4 5 6 8 9 8 10 10 11 10 12 10 13	11	Costul minim se obţine, de exemplu, pentru alegerea celor **3** străzi între aşezările **5-7, 7-8, 8-10** şi a sălii de şedinţe a Senatului în aşezarea **8** (după cum este evidenţiat în desen). Există şi alte alegeri pentru care se obţine soluţia **11**.



Descrierea soluţiei

Se elimină succesiv, dintre frunzele existente la un moment dat, frunza de cost minim. Toate nodurile au costul iniţial 1. La eliminarea unei frunze, se incrementează cu costul fiului costul tatălui acesteia. Validitatea metodei rezultă din observaţia că, la eliminarea unei frunze oarecare, tatăl acesteia poate deveni frunză la rândul lui, dar cu un cost strict mai mare decât al frunzei eliminate.

Se poate reţine arborele cu ajutorul listelor de adiacenţă (liniare sau organizate ca arbori de căutare), iar frunzele se pot memora într-un minheap de costuri, structură care se actualizează în timp logaritmic.

Rezolvare

```
#include <fstream.h>
struct NOD{int nod;NOD* next;};
NOD *dp[10000];
```

```
long s; int *c,n,k,h[10001],lh;
void add(NOD*& p, int a)
{NOD* q;
 q=new(NOD);q->next=p;
 q->nod=a;p=q;
}
void remove(NOD*& p, int a)
{NOD *q,*r;
 if (p->nod==a){
     q=p;p=p->next;
     delete q;
 }
 else {
    q=p;
    while (q->next->nod!=a) q=q->next;
    r=q->next;q->next=q->next->next;
    delete r;
 }
}
void citire()
{int i,x,y;
 ifstream f("cezar.in");
 f>>n>>k;
 c=new int[10000];
 for (i=0;i<n;i++){
   *(c+i)=1;dp[i]=0;h[i]=n;
 }
 for (i=0;i<n-1;i++){
   f>>x>>y;
   add(dp[x-1],y-1);add(dp[y-1],x-1);
 }
 for (i=0;i<n;i++)
```

```
   if (!dp[i]->next)h[++lh]=i;
 *(c+n)=20000;h[n]=n;
 f.close();
}
void desfrunzire()
{ int ii,i,j,pmin;long min;
  s=0;
  for (ii=n-1;ii>=k+1;ii--){
    pmin=h[1];s+=*(c+pmin);
    j=dp[pmin]->nod;
    *(c+j)+=*(c+pmin);
    remove(dp[j],pmin);
    if(!dp[j]->next)h[1]=j;
    else {h[1]=h[lh];h[lh]=n;lh--;}
    i=1;
    while (2*i<=lh){
     if (*(c+h[2*i])<*(c+h[2*i+1])) j=2*i; else j=2*i+1;
     if (*(c+h[i])>*(c+h[j]))
     {int aux=h[i];h[i]=h[j];h[j]=aux;i=j;}
     else i=n;
    }
  }
}
void main()
{ citire();
  desfrunzire();
  ofstream f("cezar.out");
  f<<s<<'\n';
  f.close();
}
```

3. Graf (Olimpiada de Informatică, faza judeţeană 2006)

Se ştie că într-un graf neorientat conex, între oricare două vârfuri există cel puţin un lanţ iar lungimea unui lanţ este egală cu numărul muchiilor care-l compun. Definim noţiunea *lanţ optim între două vârfuri X şi Y* ca fiind un lanţ de lungime minimă care are ca extremităţi vârfurile X şi Y. Este evident că între oricare două vârfuri ale unui graf conex vom avea unul sau mai multe lanţuri optime, depinzând de configuraţia grafului.

Cerinţă:

Fiind dat un graf neorientat conex cu N vârfuri etichetate cu numerele de ordine 1,2,…,N şi două vârfuri ale sale notate X şi Y ($1 \le X,Y \le N$, $X \ne Y$), se cere să scrieţi un program care determină vârfurile care aparţin **tuturor** lanţurilor optime dintre X şi Y.

Date de intrare:

Fişierul **graf.in** conţine

- pe prima linie patru numere naturale reprezentând: N (numărul de vârfuri ale grafului), M (numărul de muchii), X şi Y (cu semnificaţia din enunţ).
- pe următoarele M linii câte două numere naturale nenule A_i,B_i ($1 \le A_i,B_i \le N$, $A_i \ne B_i$, pentru $1 \le i \le M$) fiecare dintre aceste perechi de numere reprezentând câte o muchie din graf.

Date de ieşire:

Fişierul **graf.out** va conţine

- pe prima linie, numărul de vârfuri comune **tuturor** lanţurilor optime dintre X şi Y;
- pe a doua linie, numerele corespunzătoare etichetelor acestor vârfuri, dispuse **în ordine crescătoare**; între două numere consecutive de pe această linie se va afla câte un spaţiu.

Restricţii:

- $2 \le N \le 7500$; $1 \le M \le 14000$
- pentru 50% din teste $N \le 200$

Exemple:

graf.in	graf.out	graf.in	graf.out
6 7 1 4	3	3 2 1 3	2
1 2	1 4 5	1 2	1 3
1 3		3 1	
1 6			
2 5			
3 5			
5 6			
5 4			

Descrierea soluţiei

Dacă definim distanţa între două vârfuri ale unui graf neorientat ca fiind lungimea celui mai scurt lanţ dintre care are drept capete vârfurile, atunci putem să observăm că un vârf oarecare Z se află pe un lanţ de lungime minima dintre X şi Y dacă şi numai dacă d(X,Z)+d(Z,Y)=d(X,Y) , pentru cazul în care considerăm lungimea lanţului ca fiind numărul muchiilor şi d(X,Z)+d(Z,Y)=d(X,Y)+1, pentru cazul în care considerăm lungimea ca fiind numărul vârfurilor.

Stabilim prin câte o parcurgere în lăţime distanţele tuturor vârfurilor faţă de X şi respectiv Y (capetele lanţului, citite din fişier). Vedem care dintre vârfurile ce aparţin cel puţin unui lanţ de lungime minima între X şi Y au proprietatea ca sunt singurele aflate la o anumită distanţă de X. Acestea sunt vârfurile care aparţin tuturor lanţurilor de lungime minimă dintre X şi Y.

Rezolvare

```
#include <fstream.h>
const nmax = 7501;
struct node { int x; node * next;};
typedef int sir[nmax];
```

```
typedef node* nod;
nod v[nmax];
int *dx, *dy, *num, *vertex, *at, l[nmax];
int n,m,x,y,i,j,xx,yy,nr;
nod p;
ifstream fi("graf.in");
void bfs (int x, int d[])
{ int tail, ttail, head, i;
  nod p,q;
  setmem(d,(n+1)*sizeof(int),0);
  head=1; tail=1; ttail=1;
  l[1]=x; d[x]=1;
  do
  { for (i=head; i<=tail; i++)
    {
     p = v[l[i]];
     while (p!=NULL)
      { if (d[p->x]==0)
            { ttail++;
              l[ttail] = p->x;
              d[p->x] = d[l[i]]+1;
             }
             p = p->next;
       }
     }
    if (tail==ttail) break;
    head = tail+1;
    tail = ttail;
    }
  while (1);
}
```

```
void main()
{
 fi>>n>>m>>x>>y;
 for (i=1;i<=m;i++)
   { fi>>xx>>yy;
     p = new node;
     p->x = yy;
     p->next = v[xx];
     v[xx] = p;
     p = new node;
     p->x = xx;
     p->next = v[yy];
     v[yy] = p;
   }
 dx = new int[n+1];
 bfs(x, dx);
 dy = new int[n+1];
 bfs(y,dy);
 num = new int[n+1];
 for (i=0;i<=n;i++)
   num[i]=0;

 vertex = new int[n+1];
 for (i=0;i<=n;i++)
   vertex[i]=0;
 for (i=1;i<=n;i++)
   if (dx[i] + dy[i] == dx[y]+1)
     { num[dx[i]]++;
       vertex[dx[i]] = i;
     }
 at = new int[n+1];
```

```
for (i=0;i<=n;i++)
  at[i]=0;
nr = 0;
for (i=1;i<=n;i++)
  if (num[i]==1)
    { nr++;
      at[vertex[i]]=1;
    }
ofstream fo("graf.out");
fo<<nr<<endl;
for (i=1;i<=n;i++)
  if (at[i]==1)
   fo<<i<<' ';}
```

3.3.2. Probleme propuse spre rezolvare

1. Problema sponsorului (Olimpiada Naţională de Informatică, Suceava 1996) RATUC Suceava, unul dintre sponsorii olimpiadei, îşi propune să îmbunătăţească transportul în comun în localitate. Directorul va pune la dispoziţie o schemă pe care sunt reprezentate staţiile, numerotate pentru simplificare, de la 1 la n şi cele k linii directe între staţii, astfel încât între oricare două staţii există legătură, eventual cu schimbarea mijlocului de transport. Trebuie să determinaţi dacă există cel puţin o linie directă prin blocarea căreia legătura, directă sau indirectă, între cel puţin două staţii să fie întreruptă. Dacă astfel de linii există, să se propună înfiinţarea unui număr cât mai mic de linii directe între staţiile existente astfel încât prin blocarea unei singure linii directe, oricare ar fi aceasta, circulaţia între oricare două staţii să fie posibilă; se alege soluţia pentru care suma ocolurilor pe traseele varianta (măsurate în număr de linii directe) să fie cât mai mică. Implementaţi eficient algoritmul lui Kruscal de determinare a unui arbore Speologii au cercetat n culoare subterane pentru a stabili dacă aparţin aceleiaşi peşteri. Prin tehnici

specifice de curenţi de aer şi de colorare a cursurilor de apa, a fost demonstrată existenţa unor canale de legătură între mai multe culoare. Precizându-se perechile de culoare între care au fost stabilite legături, să se afle dacă sistemul de culoare corespunde unei singure peşteri .

2. Într-o staţiune de tratament s-au întâlnit n sahişti, codificaţi prin numere de la 1 la n, care au jucat p partide de şah. Pentru fiecare partidă se precizeaza cei doi jucători. Să se determine un grup cât mai mare de jucători care, prin jocurile pe care le-au desfăşurat, au format o grupare de tip turneu (în care fiecare a jucat cu fiecare).

3. Reţeaua de distribuţie a apei calde pentru o centrala termica zonală este formată dintr-un sistem de conducte care leagă centrala de blocuri şi blocurile între ele. Blocurile din reţea sunt numerotate cu numere întregi de la 1 la n, centrala fiind punctul 0 de distribuţie. Se cunosc distanţele de la centrală la blocuri precum şi distanţele între oricare doua blocuri . Să se afişeze perechile de numere desemnând punctele de distribuţie între care trebuie să se monteze conducte astfel încât fiecare bloc sa fie alimentat cu apă caldă (nu neapărat direct de la centrală) şi lungimea totală a conductelor necesare să fie minimă .

4. Un elev vrea să călătorească din localitatea X în localitatea Y . Dacă în ţara respectivă există n localităţi şi ştiind timpul necesar pentru a ajunge dintr-o localitate în alta (în cazul în care se poate ajunge direct) se cere să se determine timpul minim în care elevul poate să ajunga din X in Y .

5. Se consideră un grup de n persoane, având fiecare un anumit număr de preocupări, cel mult p preocupări fiecare . Preocupările sunt codificate prin numere întregi nenule.

Să se determine perechile de persoane care au preocupări comune, specificând pentru fiecare astfel de pereche numărul de preocupări comune .

Să se determine perechea (eventual perechile) de persoane cu un număr maxim de preocupări comune.

Să se determine preocuparea îmbrăţişată de un număr maxim de persoane.

6. Într-o zona de munte, există n cabane, între cabane existând trasee de legătură . Să se determine, dacă există, o ordine de vizitare a cabanelor, astfel încât să se parcurgă o

singura dată toate traseele de legătură din zonă revenind la aceeaşi cabană de la care s-a pornit .

7. Fie un arbore cu n vârfuri . În fiecare nod al arborelui se află câte un bec . Prin atingerea unui bec acesta îşi schimbă starea (din aprins în stins şi invers), la fel şi vecinii lui directi. Considerând că iniţial toate becurile sunt stinse, să se scrie un program care generează o secvenţă de „atingeri" prin care pomul este aprins complet. Datele de intrare se citesc din fişierul „pom.in" cu formatul : pe prima linie numărul de noduri , şi pe urmatoarele n-1 linii muchiile arborelui .

8. Să presupunem că un graf neorientat este format din n vârfuri şi m arce. Se stie că sunt folosite toate muchiile pentru a lega vârfurile şi că doua vârfuri pot fi legate direct prin cel mult o muchie. Să se determine valoare maximă a expresiei $S = \sum_{i=1}^{n} d(i)^2$, unde d(i) reprezintă gradul nodului i .

9. Într-un oraş intersecţiile sunt numerotate de la 1 la n (se consideră intersecţie nu numai locul unde se intersectează mai multe străzi ci şi capetele de străzi). Edilii oraşului doresc să numeroteze şi strazile oraşului, dar într-un mod care să tină seama de numerotarea intersecţiilor, şi anume: doua străzi diferite să aiba numere diferite şi în fiecare intersecţie trebuie să sosească o stradă care să aibă numărul intersecţiei. În condiţiile acestei probleme, prin stradă se înţelege o porţiune de drum aflată între doua puncte de intersecţie, neglijând faptul că în practică străzile cuprind mai multe intersecţii. Întrebarea este dacă se poate , şi dacă da , cum se poate face o astfel de numerotare .

În fişierul de intrare se găsesc mai multe seturi de date separate de cate un rând liber; un set de date are pe prima linie numărul n $(n \le 150)$ de puncte de intersecţie(terminale). Fiecare astfel de punct este identificat prin numărul cu care este numerotat, iar pe următoarele n linii sunt date pentru fiecare intersecţie k $1 \le k \le n)$ puncte cu care acesta comunică direct printr-o stradă .

Pentru fiecare set de date trebuie afişată o numerotare a străzilor dacă există, respectiv mesajul: „Nu există o astfel de numerotare" dacă nu există . Numerotarea se afişează astfel: pe fiecare linie o pereche de numere care reprezintă intersecţiile care delimitează strada şi apoi numărul străzii .

CONCLUZII

Metodica predării informaticii este o disciplină care diferă de disciplinele propriu-zise de informatică în conţinut şi stil, dar diferă şi de metodica predării altor discipline (de exemplu, matematică). Ea are legătură cu alte ştiinţe, după cum urmează:

- ***Pedagogia*** adică ştiinţa care se ocupă cu studiul metodelor de educaţie şi de instruire a oamenilor, în special a persoanelor cu puţina experienţă.
- ***Psihologia*** este ştiinţa care se ocupă cu studiul proceselor şi particularităţilor psihice umane.
- ***Metodica*** este partea didacticii generate care studiază principiile, metodele şi formele de predare adaptate specificului fiecărui obiect de învăţământ.
- ***Didactica*** este o parte a pedagogiei care se ocupă cu principiile şi metodele predării materiilor de învăţământ, precum şi cu organizarea învăţământului.

Din punctul de vedere al unui cadru didactic, aceste ştiinţe sunt importante în egală măsură şi trebuie studiate/stăpânite simultan. Cunoştinţele acumulate (oricât de vaste şi de profunde ar fi) nu sunt suficiente pentru desfăşurarea procesului de instruire. Pentru ca activitatea profesorului să aibă rezultatul dorit este nevoie de un mediu corespunzător (legislativ, economic, social, etc**),** dar şi de talent şi perseverenţa.

Începând cu 1980, termenul „informatică" a fost un sinonim pentru: ştiinţa calculului, ştiinţa calculatoarelor, ingineria calculatoarelor, tehnologia informaţiei şi a comunicării ş.a.m.d. Aceste definiţii informale pot căpăta noi valenţe, sub o formă mai mult sau mai puţin detaliată, fără însă a avea pretenţia că sunt complete. Iată doar câteva posibile exemple:

1**.** ***Informatica*** se ocupă cu studiul calculatoarelor şi al fenomenelor majore din jurul acestora.

2. ***Informatica*** cuprinde totalitatea cunoştinţelor asupra calculatorului şi al calculului.

Ea are componente teoretice, experimentale şi de proiectare şi include:

a) teorii pentru înţelegerea echipamentelor de calcul, a programelor şi sistemelor;

b) experimente pentru testarea şi dezvoltarea conceptelor;

c) metodologii (reguli, metode) de proiectare (algoritmi, unelte pentru aplicaţii practice particulare),

d) metode de analiză pentru verificarea faptului că realizările îndeplinesc cerinţele.

3. ***Informatica*** se ocupă cu studiul reprezentării cunoştinţelor si a implementării acestora.

4. ***Informatica*** se ocupă cu studiul modelelor si a complexităţii cunoştinţelor.

5. ***Informatica*** se ocupă cu studiul sistematic al proceselor algoritmice care descriu şi transformă informaţia (teoria informaţiei), precum şi cu analiza, proiectarea, implementarea şi aplicarea acestora.

Vom admite astfel că scopurile introducerii informaticii ca disciplină de sine stătătoare (opţională/facultativă/obligatorie) în planurile de învăţământ şcolare sunt:

- Dezvoltarea capacităţii de utilizare a terminologici, a unui limbaj informatic specific si de folosire a tehnicii de calcul în însuşirea de noi cunoştinţe.
- înţelegerea informaticii ca mijloc de modelare a fenomenelor realităţii înconjurătoare şi de simulare a acestora.
- Asigurarea nivelului de cultură generală în informatică prin parcurgerea principalelor etape prin dezvoltarea informaticii ca ştiinţă.
- Dezvoltarea unei motivaţii intrinseci în studiul informaticii.
- Crearea unei atitudini favorabile activităţii de rezolvare a problemelor cu ajutorul calculatorului, prin deprinderea strategiilor de abordare a acestora si tratarea lor într-un mod riguros.
- Dezvoltarea unor capacităţi de autoinstruire.
- Crearea unei atitudini pozitive privind importanţa deosebită a informaticii în lumea contemporană şi pătrunderea ei în toate domeniile vieţii economico-sociale.

BIBLIOGRAFIE

1. **C. Bălcău,** *Combinatorică şi teoria grafurilor*, Editura Universităţii din Piteşti, Piteşti, 2007.

2. **E. Cerchez, M. Şerban,** Programarea în Limbajul C / C++ pentru liceu, POLIROM, 2006;

3. **E. Ciurea**, Algoritmi. Introducere în algoritmica grafurilor, Editura Tehnică, Bucureşti, 2001;

4. **T.H. Cormen , C.E. Leiserson , R.R. Rivest,** *Introducere in Algoritmi*, Computers Libris Agora, 2001;

5. **L. Ezechil,** *Prelegeri de didactica generala*, Editura Paralela 45 Educational, Piteşti 2003 ;

6. **D.E. Knuth,** Tratat de programarea calculatoarelor. Algoritmi fundamentali, Editura Tehnică , Bucureşti , 1974;

7. **D. Logofătu**, Algoritmi fundamentali în C++. Aplicaţii, POLIROM, 2007;

8. **G.D. Mateescu, P.F. Moraru, C. Popescu**, *Informatică. Teste şi variante de subiecte pentru bacalaureat*, Editura Donaris, Sibiu, 2001;

9. **O. Pătrăşcoiu, Gh. Marian, N. Mitroi,** *Elemente de grafuri şi combinatorică. Metode, algoritmi şi programe*, Editura All Bucureşti, 1994;

10. **C. Petre, D. Popa, Şt. Crăciunoiu, C. Iliescu,** *Metodica predării Informaticii şi Tehnologiei Informaţiei*, Editura Arves, Craiova, 2002;

11. **R. Pintea, D. Popescu Anastasiu,** *Bacalaureat la Informatică,* Editura L&S Infomat, Bucureşti, 2005;

12. **I. Tomescu,** *Probleme de combinatorică şi teoria grafurilor,* Editura Didactică şi Pedagogică, Bucureşti, 1981.