ABORDĂRI DIDACTICE COMPLEXE PRIVIND PREDAREA-ÎNVĂȚAREA TEHNICII DIVIDE ET IMPERA PRIN INTERMEDIUL TABLEI INTERACTIVE

Angela GLOBA, UST

angelagloba@gmail.com

Rezumat. *Articolul respectiv conține abordări metodice care ţin de utilizarea tablei interactive în procesul de predare-învăţare a tehnicii Divide et Impera, în situația unor cazuri complexe. Se scoate în evidență rolul benefic al tablei interactive în procesul de asimilare rapidă a noilor concepte de către studenţi. Sunt examinate aspecte didactice prin prisma aplicării SMART Notebook-ului în procesul de predare-învăţare a tehnicilor de programare. Se analizează în mod special rolul creator în procesul didactic şi imaginația de care trebuie să dispună un profesor pentru a dezvolta gândirea logica a studenţilor, pentru a încuraja participarea lor activă şi a le crea deprinderi de sistematizare şi generalizare a celor învăţate.*

1. Noi paradigme în procesul de predare – învăţare - evaluare

Învăţarea academică este o interacţiune complexă între cel care învaţă, materialele de instruire, repertoriul de strategii de învăţare disponibile şi contextul învăţării, care include profesorul.

Conform cercetărilor recente, învăţarea este un proces activ, cognitiv, contructiv, semnificativ, mediat şi autoreglat, [1]. „Învăţarea nu este ceva care li se întâmplă studenţilor; este ceva care se întâmplă prin studenţi”, [2].

Conform lui Zimmerman, ilustru teoretician şi promotor al autoreglării învăţării, pentru ca învăţarea să aibă loc, studenţii trebuie să se angajeze activ în propriile procese de învăţare. Majoritatea cercetătorilor accentuiază asupra faptului că autoreglarea învăţării antrenează studenţii să-şi fixeze un scop al învăţării, să se implice în acţiuni de autodirecţionare spre reliazarea ţelului pus, să-şi monitorizeze comportamentele de învăţare, ajustându-le astfel ca să le asigure succesul, [3].

Reeşind din acest context, autorul afirmă că, implicarea cât mai activă a studenţilor în procesul de predare-învăţare, stabilirea unui feedback pozitiv, îi ajută pe studenţi să devină conştienţi de propria gândire, să-şi direcţioneze motivaţia spre scopuri academice valoroase, să înveţe a fi propriul său învăţător.

În procesul de predare – învăţare - evaluare a obiectului „Tehnici de programare”, studenţii, nu o singură dată, în timpul lucrului inividual sau a orelor de laborator, au cerut să fie afişate pe tablă notiţele făcute de profesor la tema respectivă, fapt menţionat de ei în chestionar. Aceasta motivează şi mai mult aplicarea cu succes a tablei interactive în procesul de predare – învăţare – evaluare, [4]. Tindem să credem că, problemele propuse în continuare confirmă, încă o dată, necesitatea şi rolul benefic al tablei interactive în procesul de asimilare rapidă a noilor concepte de către studenţi.

În continuare vom examina, din punct de vedere metodic, 3 probleme soluționarea cărora se va face prin aplicarea unor strategii didactice noi care presupun aplicarea tablei interactive.

Problema 1: ***Algoritmul de căutare binară.*** *Se dă un şir de* ***n*** *numere întregi ordonate crescător. Să se determine dacă şirul conţine valoarea dată* ***x.***

Din propria experienţă rezolvarea acestei probleme trebuie începută prin soluționarea unui caz particular. Rezolvând exemplul propus, pas cu pas, studentului i se oferă posibilitatea de a înţelege, mai uşor noile noţiuni, de a ajunge singur la concluziile necesare. Implicarea activă în procesul de descoperire a noilor concepre permite formarea de competenţe durabile, mult mai stabile în spaţiu şi în timp.

Utilizând tabla interactivă şi instrumentele puse la dispoziţia profesorului de soft-ul SMART Notebook situaţia creată poate fi eficient dirijată [5]. Studentul este ghidat spre răspunsuri şi concluzii corecte de profesor prin utilizarea fişelor colorate, a măştii de ecran, a afectelor de animaţie şi, nu în ultimul rând, de notaţiile făcute cu cerneală digitală pe tablă. De exemplu, pentru şirul de numere A=(-4, -2, 0, 1, 3, 5, 6, 8, 9, 11, 14) şi x=8 vom avea:

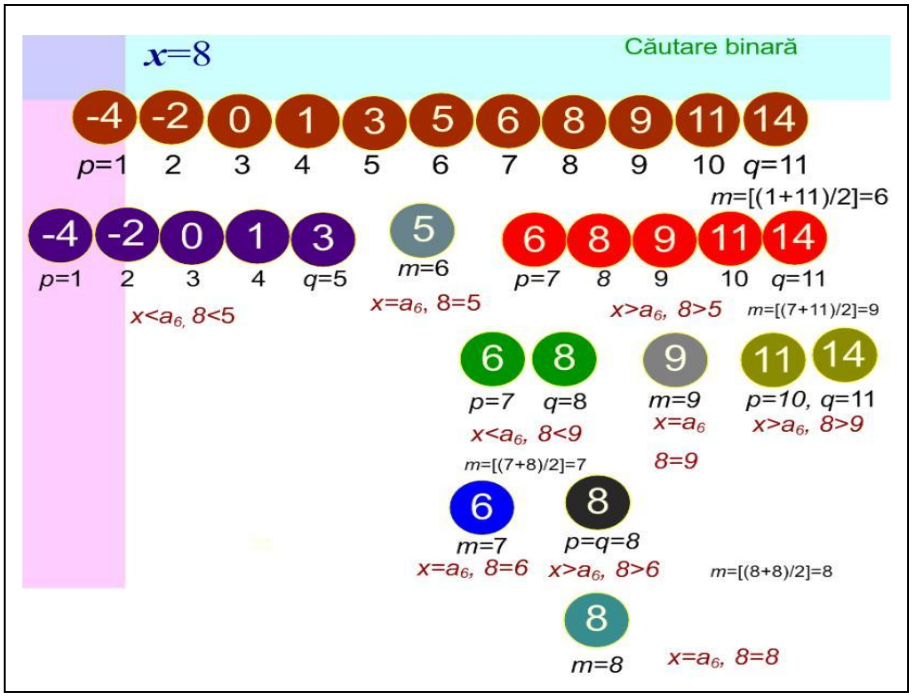

Figura 1. Algoritmul de căutare binară utilizând tehnica Divide et Impera.

Rezolvând acest exemplu, studenţii sunt ghidaţi să caute răspuns la următoarele întrebări – cheie:

- Ce tehnică se aplică pentru rezolvarea acestei probleme? (R: Divide et Impera)
- În cîte subprobleme se divide problema iniţială? (R: în trei)

- Care sunt dimensiunile acestor subprobleme? (R: două subprobleme au dimensiunea (n-1)/2, iar o problemă are dimensiunea 1)
- Care din aceste subprobleme este elementară? (R: cea de dimensiunea 1)
- În calitate de subprogram vom utiliza o procedură sau o funcţie? (R: o funcţie de tip logic - *Cautare*)
- Care vor fi parametrii subprogramului dat? (R: vom avea doi parametri – p,q, care indică indicele primului element şi a ultimului elemenet din şir)
- Vom utiliza o implementare iterativă sau recursivă a subprogramului dat? (R: recursivă)
- Care este definirea recursivă a *Algoritmului de căutare binară*?

$$Cautare(p,q)=\begin{cases} false, q<p \\ true, x=a[mij], mij=\left[\frac{p+q}{2}\right] \\ Cautare(p, mij-1), x<a[mij] \\ Cautare(mij+1, q), x>a[mij] \end{cases}$$

- Cum este înfăptuită faza de asamblare a soluţiilor subproblemelor? (R: nu există, deoarece soluţia unei subprobleme reprezintă soluţia problemei iniţiale.)
- Scrieţi subprogramul care implementează *Algoritmul de căutare binară*.

```
Function Cautare(p,q:integer):boolean;
var mij:integer;
begin
 if q<p then Cautare:=false
 else
 begin
 mij:=(p+q) div 2;
 if x=a[mij] then Cautare:=true
  else
  begin
  if x<a[mij] then Cautare:=Cautare(p,mij-1);
  if x>a[mij] then Cautare:=Cautare(mij+1,q);
  end;
 end;
end;
```

În continuare să examinăm următoarea problemă.

Problema 2. ***Sortarea prin metoda QuickSort.*** *Se dă un şir de* **n** *numere întregi. Să se ordoneze crescător acest şir.*

Sortarea prin această metodă are la bază următorul principiu: tabloul este ordonat în aşa fel, încât pentru fiecare element din tablou $x[poz]$, elementele situate la stânga sunt mai

mici decât *x*[*poz*], iar cele din dreapta sunt mai mari sau egale decât *x*[*poz*]. Sortarea tabloului *x* decurge astfel:

- se prelucrează o secvenţă din vector cu indici cuprinşi între *p* şi *q;*
- se ia una din aceste componente, fie *piv*=*x*[*p*], care se consideră element pivot;
- în interiorul tabloului se fac interschimbări de componente, astfel încât toate cele mai mici decât valoarea pivot să treacă în stângă a acesteia, iar elementele cu valoare mai mare decât pivot să treacă în dreapta; prin această operaţie se va deplasa şi valoarea pivot, astfel că ea nu se va mai găsi pe poziţia iniţială ci pe o poziţie corespunzătoare relaţiei de ordine. Fie aceasta este notat prin *k*.
- ***Atenţie***! În urma acestui set de operaţii elementele din stânga sunt mai mici decât pivot, dar nu neapărat şi în ordine. La fel cele din dreapta sunt mai mari, dar nu neapărat şi în ordine;
- se continuă setul de operaţii similare, aplicând recursiv metoda pentru zona de tablou situată în stânga componentei pivot şi pentru cea din dreapta acesteia;
- oprirea recursiei se face când lungimea zonei de tablou care trebuie sortată devine egala cu unitate.

Metoda de sortare QuickSort este o metodă mai dificil de înţeles, deşi este una din cele mai rapide metode de sortare. Utilizarea a doi indici care tot timpul îşi modifică valorile şi a unei valori *pivot* este nu tocmai simplu de asimilat. În astfel de cazuri, rezolvarea mai multor exemple concrete poate ajuta studentul să înţeleagă mai rapid această metodă. La rezolvarea şi exemplificarea acestei probleme se va utiliza aşa o opţiune a soft-ului SMARTNotebook pentru tablele interactive cum este *clonarea obiectelor*. Iniţial pe tablă se va afişa şirul de numere *X*=(6, 9, 4, 8, 0, 1, 5, 3, 2, 7) şi marcatorii *p* -  şi *q* -  . Profesorul va explica, pe baza exemplului propus, în ce constă metoda de sortare QuickSort. Pentru aceasta:

- va clona şirul de numere iniţial şi va efectua prima parcurgere;
- se vor evidenţia care sunt subproblemele problemei iniţiale;
- se va aplica algoritmul QuickSort pentru fiecare din subproblemele obţinute;
- subprobleme obţinute în faza finală vor indica şirul de numere iniţial sortat crescător;
- se va cere formularea concluziei referitor la faza de asamblare, care nu există, deoarece şirul obţinut în final este cel sortat căutat.

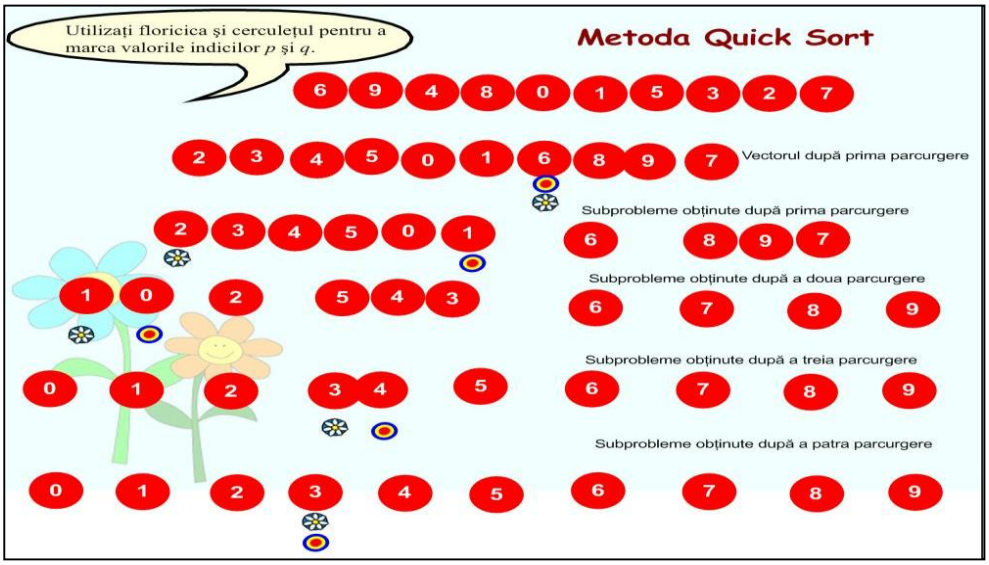

Figura 2. Sortarea prin intermediul metodei QuickSort aplicând Divide et Impera

Să examinăm o soluţie de implementare. Funcţia *poz* verifică o porţiune din vector cuprinsă între indicii *p* şi *q; piv*=*x*[*p*] este un reper. La sfîrşitul execuţiei acestei funcţii *piv* se va găsi pe o poziţie *k* cuprinsă între *p* şi *q* astfel încât toate componentele vectorului cuprinse între *p* şi *k*-1 vor fi mai mici decât *piv* iar componentele cuprinse între *k*+1 şi *q* vor fi mai mari decât *piv*. Deci *piv* se va găsi pe pozitia *k* corespunzătoare ordinii în vector. Complexitatea algoritmului QuickSort este $O(n*\log(n))$**.**

2. Examinarea unor probleme cu un grad avansat de dificultate

Scopul oricărei activităţi de predare este de a înarma studenţii cu un sistem de cunoştinţe armonios şi corect din punct de vedere ştiinţific. Logica internă a disciplinei şi legile generale ale dezvoltării capacităţilor de cunoaştere individuale impun asigurarea continuităţii, dar şi necesitatea sistematizării materiei. Noile informaţii relevante vor fi legate de cele deja introduse şi vor prefigura informaţiile ulterioare. Baza principiului sistematizării cunoştinţelor se concretizează prin expuneri organizate asupra cunoştinţelor de asimilat, respectându-se un anumit plan. Pentru a dezvolta continuu gândirea logica a studenţilor, pentru a încuraja participarea lor activa, pentru a le crea deprinderi de sistematizare şi generalizare a celor învăţate, profesorul trebuie să-şi folosească la maximum disponibilităţile creatoare şi talentul pedagogic în pregătirea materialelor pentru lecţie. Activitatea individuală conştientă a studentului ar trebui să fie esenţială. Cunoştinţele nu se pot asimila în salturi, iar deprinderile neexersate se pierd. Dacă dorim un învăţământ de masă eficient şi asigurarea unei pregătiri ritmice a studenţilor, trebuie să se accepte şi un control permanent şi riguros al profesorului asupra modului şi stadiului de însuşire a cunoştinţelor de către studenţi.

Educaţia are misiunea de a crea competenţe, capacităţi, abilităţi şi deprinderi, care vor permite o adaptare rapidă a individului la situaţiile de viaţă curente cu care va fi

confruntat. Fixarea comportamentelor nu implică, totuşi, în mod automat, o exigenţă de repetare mecanică. Funcţia de stabilizare a cunoştinţelor este strâns legată de cea de exersare, dar nu cu cea de repetare automată a celor învăţate. Cum să evităm învăţarea „pe de rost" şi exersarea strict mecanică? Modelul lui Tirtiaux [6] dă un răspuns la aceste întrebări, propunând o gradaţie calitativă a unor exerciţii funcţionale:

- a reproduce cele învăţate - este activitatea în care individul trebuie să rezolve încă o dată, dar singur, cazurile tratate în situaţiile de învăţare;
- a recunoaşte conceptele învăţate - este activitatea prin care individul trebuie să distingă, într-o serie de exemple pe care le examinează de unul singur şi pentru prima oară, pe cele a căror structură este analoagă structurii de învăţare şi pe cele a căror structură este diferită;
- ajustarea celor învăţate la noile condiţii - este o activitate în care individul încearcă să încadreze o anumită problemă (sarcină) în condiţiile unui exemplu sau concept studiat anterior;
- aplicarea noţiunilor învăţate - este activitatea în care individul, după după ce a luat cunoştinţă în mod clar de noţiune, o utilizează în cadrul unor aplicaţii mai complexe, eventual propuse de profesor sau dictate de viaţă.

Problema propusă în continuare respectă toate etapele modelului propus de Tirtiaux.

Problema 3. ***Foaia de tablă.*** *Se dă o foaie de tablă de dimensiune dreptunghiulară. Foaia respectivă conţine* **n** *găuri. Se cere de tăiat din foaia dată o bucată de tablă de arie maximă, care să nu aibă găuri. Se cunosc coordonatele diagonalei foii de tablă* ($x1$,$y1$) *şi* ($x2$,$y2$). *Coordonatele găurilor identificate sunt numere intregi. Sunt permise numai tăieturi orizontale şi verticale.*

Rezolvarea acestei probleme implică următoarele cunoştinţe elementare: coordonate carteziene în plan; aria unui dreptunghi; aflarea maximului dintre două numere; probleme elementare şi neelementare, tehnica Divide et Impera; definirea subproblemelor unei probleme; recursie ş.a.

După actualizarea acestor cunoştinţe studenţii vor trebui să răspundă la întrebările: Ce se întâmplă în cazul când tabla nu are nici o gaură? Cu ce poate fi asociat acest caz particular? (R: se calculează aria; este o problemă elementară) Vor scrie formula de calculare a ariei unui dreptunghi definit de punctele diagonale ($x1$,$y1$) şi ($x2$,$y2$): $aria$=($x2$-$x1$)*($y2$-$y1$).

Utilizând instrumentul *Captură de ecran* profesorul poate complica problema, şi anume, prin indicarea unei găuri de coordonate (x_i,y_i). Scopul profesorului este ca studenţii să clarifice ce tehnică de programare se va utiliza, să identifice subproplemele problemei iniţiale şi să descrie care vor fi etapele algoritmului de calcul şi cum se va implementa acest algoritm: recursiv sau iterativ pentru problema *Foaia de tablă*. (Fig.3)

Analizând imaginea din figura 3 profesorul va conduce studenţii spre următoarea concluzie: *aria*=*max*(*D*1, *D*2, *D*3, *D*4), unde D_i este aria unui dreptunghi obţinut prin tăierea orizontală sau verticală a dreptunghiului iniţial (în dependenţă de coordonatele găurii).

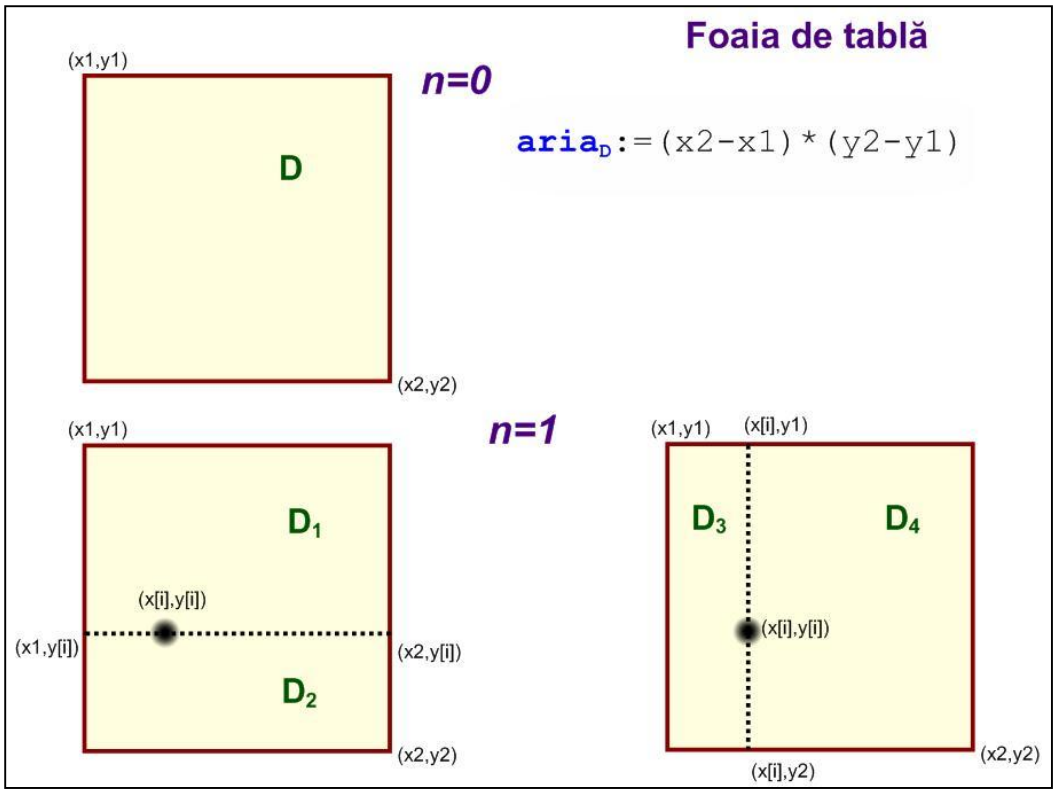

Figura 3.Problema *Foaia de tablă*, *n*=0, *n*=1, soluţionată prin intermediul metodei Divide et Impera.

Explicând şi analizând cazul *n*=1 (o singură gaură) pentru studenţi va fi mai dificil să vadă care sunt subproblemele problemei iniţiale şi unde este implementarea recursivă. Pentru a obţine rezultatul scontat, adică pentru a primi de la studenţi subproblemele problemei iniţiale şi definirea recursivă a problemei iniţiale va propune studenţilor să clarifice situaţia când tabla conţine două găuri (Fig. 4) (analog, pentru a economisi timp, va utiliza instrumentul *Captură de ecran*). Din imagini este clar, că se va efectua o nouă divizare (vertical, orizontal) pentru dreptunghiul *D*1 şi respectiv *D*4. Aşa dar:

- subproblemele problemei iniţiale sunt: problema elementară - *aria*(*x*1,*y*1,*x*2,*y*2); subprobleme neelementare - *aria*(*x*1,,*y*1,*x*2,y[*i*]); *aria*(*x*1,,*y*[i],*x*2,y2); *aria*(*x*1,,*y*1,*x*[i],y2); *aria*(*x*[i],,*y*1,*x*2,y2);
- definiţia recursivă de calculare a ariei maxime a unui dreptunghi (fără găuri) [7]:

$$aria(x1,y1,x2,y2)=\begin{cases}(x2-x1)\cdot(y2-y1), n=0\\ \max\{\max[\ aria(x1,y1,x2,y[i]), aria(x1,y[i],x2,y2)],\\ \max[\ aria(x1,y1,x[i],y2), aria(x[i],y1,x2,y2)]\}, n>0,\\ (x[i],y[i])-coord.gaurii\end{cases}$$

unde ***n*** numărul de găuri.

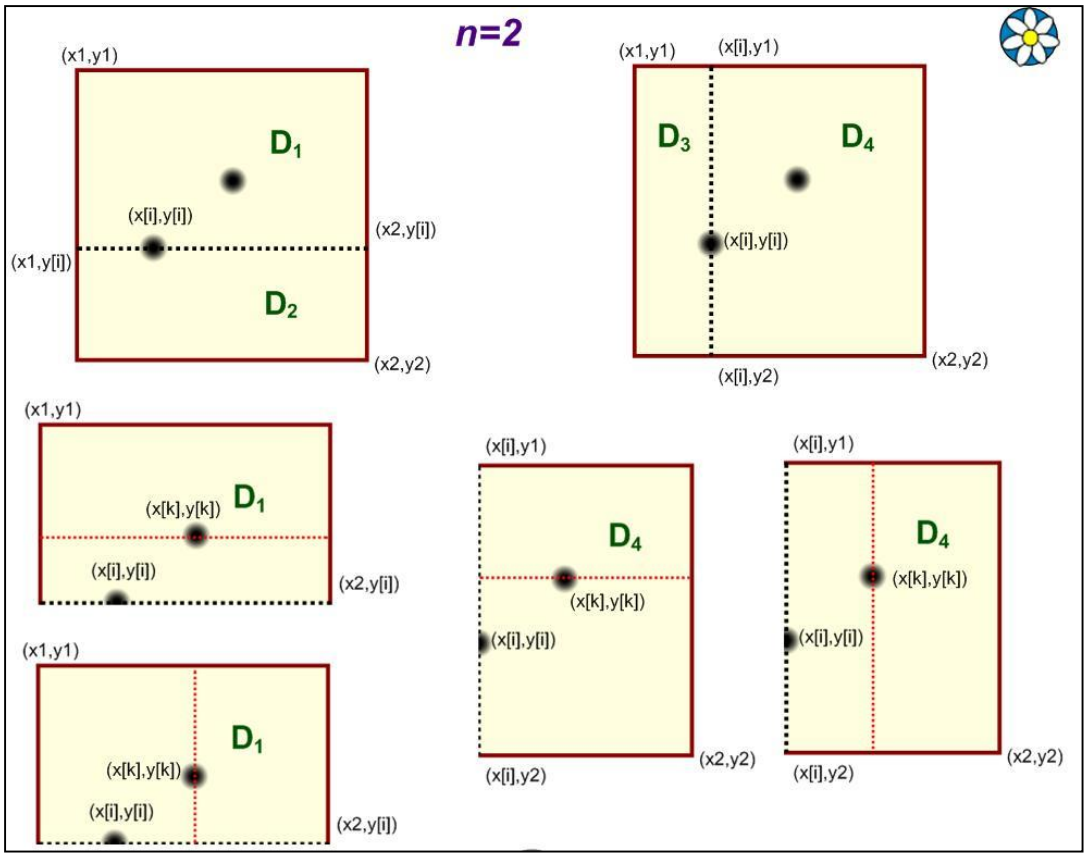

Figura 4. Problema *Foaia de tablă*, *n*=2, soluționată prin intermediul metodei Divide et Impera.

Secvenţa de program ce implementează algoritmul descris mai sus este:

```
type vector=array [1..50] of integer;
var x,y:vector; n,i:integer;
Function max(i,j:integer):integer;
begin
 if i>j then max:=i else max:=j;
end;
Function arie(x1,y1,x2,y2:integer):integer;
var i:integer; nl:boolean;
begin
 i:=1; nl:=false;
```

```
repeat
nl:=(x1<x[i]) and (x[i]<x2) and (y1<y[i]) and (y[i]<y2);
i:=i+1;
until nl or (i=n+1);
i:=i-1;
if not nl then arie:=((x2-x1)*(y2-y1)) else
arie:=max(max(arie(x1,y1,x2,y[i]),arie(x1,y[i],x2,y2)),
max(arie(x1,y1,x[i],y2),arie(x[i],y1,x2,y2)));
end;
```

Teoretic, starategia didactică ia sfârşit cu o constatare de învăţare pozitivă: a fost atins obiectivul fixat, a fost achiziţionată noua competenţă, a intervenit modificarea comportamentală, dar nu înainte de a puncta anumite concluzii referitoare la noul concept studiat. Concluziile date vor fi iniţial acoperite cu *masca de ecran* pentru a fi descoperite de studenţi desinestătător.

3. Sinteze, concluzii și recomandări

La finele acestui articol putem conchide că este foarte important să se construiască un sistem activ de cunoştinţe, generat de activitatea independentă a studentului, care presupune descoperire prin învăţarea sistematică.

Astfel, studenţii trebuie să fie învăţaţi să-şi creeze propriul mod de gândire şi studiu, să-şi dezvolte capacitatea de autocontrol a învăţării, să-şi propună scopuri şi obiective, modalităţi pentru a le atinge, să-şi planifice munca şi să-şi autoevalueze progresele.

Luând în considerație abordările didactice, examinate mai sus, experiența acumulată în procesul de predarea-învățarea tehnicii Divide et Impera, prin intermediul tablei interactive, în continuare vom puncta o serie de greșeli și erori care țin de implementarea metodei respective:

Greşelile care pot apărea la realizarea şi implementarea algoritmilor bazaţi pe tehnica Divide et Impera se pot impărţi în două categorii:

- ✓ greşeli care ţin de o strategie incorect proiectată;
- ✓ greşeli care apar în etapa de programare datorită implementării defectuoase a recursivităţii;

Pornind de la presupunerea că problema căreia i se caută soluţia acceptă rezolvare prin tehnica Divide et Impera, trebuie avute în vedere următoarele situaţii care conduc la greşeli de fond ale algoritmului:

- ✓ *Descompunerea incorectă a problemei*; (sunt generate, de regulă, de obţinerea unor subprobleme care nu sunt disjuncte);

- ✓ *Identificarea greşită a dimensiunii problemei elementare*, care acceptă rezolvarea imediată;
- ✓ *Parametrii* care indică dimensiunea subproblemelor nu tind către cazul elementar;
- ✓ Programul fiind *recursiv*, dacă *nu este prezentă condiţia de oprire*, va conduce la întreruperea execuţiei programului şi afişarea mesajului de eroare: "Stack overflow" (depăşirea stivei);
- ✓ *Faza de asamblare a soluţiilor subproblemelor* nu este prezentă, iar soluţia problemei iniţiale nu se construieşte simultan cu descompunerea sa şi nici nu reprezintă soluţia vreo unei subprobleme;
- ✓ *Proiectarea greşită a fazei de asamblare a soluţiilor*.

Pentru a-i sprijini pe studenţi în formarea lor meta-cognitivă este necesar ca prin intermediul experienţei acumulate, viziunii profesionale, profesorul, în primul rând considerăm noi, trebuie să prezinte şi să cultive studenţilor modele de gândire critică şi învăţare eficientă, și în mod special să promoveze conceptul ”învățării independente”, pentru ca discipolii săi ulterior, fiind foarte motivați, să-şi asume treptat responsabilitatea de a învăţa singuri, în scopul creșterii profesionale și integrări cu succes pe piața muncii.

BIBLIOGRAFIE

1.Beltran J., Desenrrollo y tendencias actuales de la Psicologia de la instruccion, Psicologia de la instruccion: variables y processos psicos1, J. Beltran y C. Genovard (eds.), 1996.

2.Zimmerman B. J. Theories of self-regulated learning and academic achievement: An overview and analysis. In B. J. Zimmerman & D. H. Schunk (Eds.), Self-regulated learning and academic achievement: Theoretical perspectives (2nd ed., pp. 1–37). Mahwah, NJ: Erlbaum. (2001).

3.Focşa-Semionov S., Învăţarea autoreglată, Teorie şi aplicaţii educaţionale, Chişinău, Epigraf, 2010.

4.Globa A., Utilizarea tablei interactive în procesul de predare-învăţare a tehnicii Divide et Impera din cadrul cursului universitar „Tehnici de programare”. În: Univers Pedagogic Nr. 2 (46), 2015.

5.Globa A., Pavel D., SMART Notebook. Ghid de iniţiere. UST, Chişinău, 2014.

6.Minder M., Didactica funcţională: obiective, strategii, evaluare., Cartier, Chişinău, 2003.

7.Sorin T., Tehnici de programare. Manual pentru clasa a X-a, „Editura L&S Infomat”, Bucureşti, 1996.