Cuprins

1. Terminologie

Butoane definiție – 

– sunt amplasate în text, în locul unde apare necesitatea revederii termenului respectiv şi, atunci când sunt accesate, prezintă într-o fereastră detaliu, definiția termenului respectiv.

Butoane care indică obiectivele lecției respective - 

- sunt amplasate totdeauna în partea din dreapta jos a ecranului. Prin apăsarea lor, într-o fereastră detaliu se prezintă obiectivele lecției, marcate conform momentului respectiv.

Butoane de control - 

- prin apăsarea butoanelor corespunzătoare:

-  se execută animația “pas cu pas” înainte cu/fără vizualizarea codului sursă; acelaşi buton este folosit pentru trecerea de la un exemplu la altul înainte
-  se execută animația “pas cu pas” înapoi cu/fără vizualizarea codului sursă; acelaşi buton este folosit pentru trecerea de la un exemplu la altul înapoi
-  se reia aplicația de la început generând un exemplu nou
- 

se vizualizează zona de declarații a variabilelor din codul sursă

```
#include <fstream.h>
#define NMaxVf 50
#define NMaxMuchii NMaxVf*(NMaxVf-1)/2
struct Muchie {int e1, e2, cost; };
Muchie G[NMaxMuchii];
int A[NMaxVf], c[NMaxVf];
int n, m;

void Initializare()
```

Buton care permite generarea unei noi situații practice / a unui nou graf – 

– atunci când este accesat generează un nou plan.

Buton care permite generarea unui nou test – atunci când este accesat 

generează un nou test.

Buton explicație – 

– atunci când este accesat deschide/închide o fereastră detaliu care oferă explicații despre semnificația legăturilor.

Indicații privind modul de lucru cu editorul – nu sunt accesate ci doar afişate pe ecran.

Butoane selectare moment – 

– atunci când sunt accesate se realizează operația de selectare şi afişare a momentului corespunzător din obiectul de conținut curent.

Butoane selectare întrebare – 

– atunci când sunt accesate se realizează operația de selectare şi afişare a item-ului corespunzător.

Butoane selectare răspuns corect – 

– atunci când sunt accesate se realizează operația de selectare a răspunsului considerat corect.

Butoane de indicare a corectitudinii răspunsului

- răspuns corect – 

- răspuns eronat – 

Butoane de activare a codului de determinare a APM : 

activarea butoanelor, în situații diferite, au acelaşi efect – determinarea şi vizualizarea arborelui parțial de cost minim.

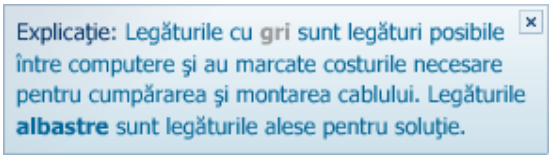
Butoane de selectare a obiectului de conținut – 

– atunci când sunt accesate realizează activarea obiectului de conținut corespunzător.

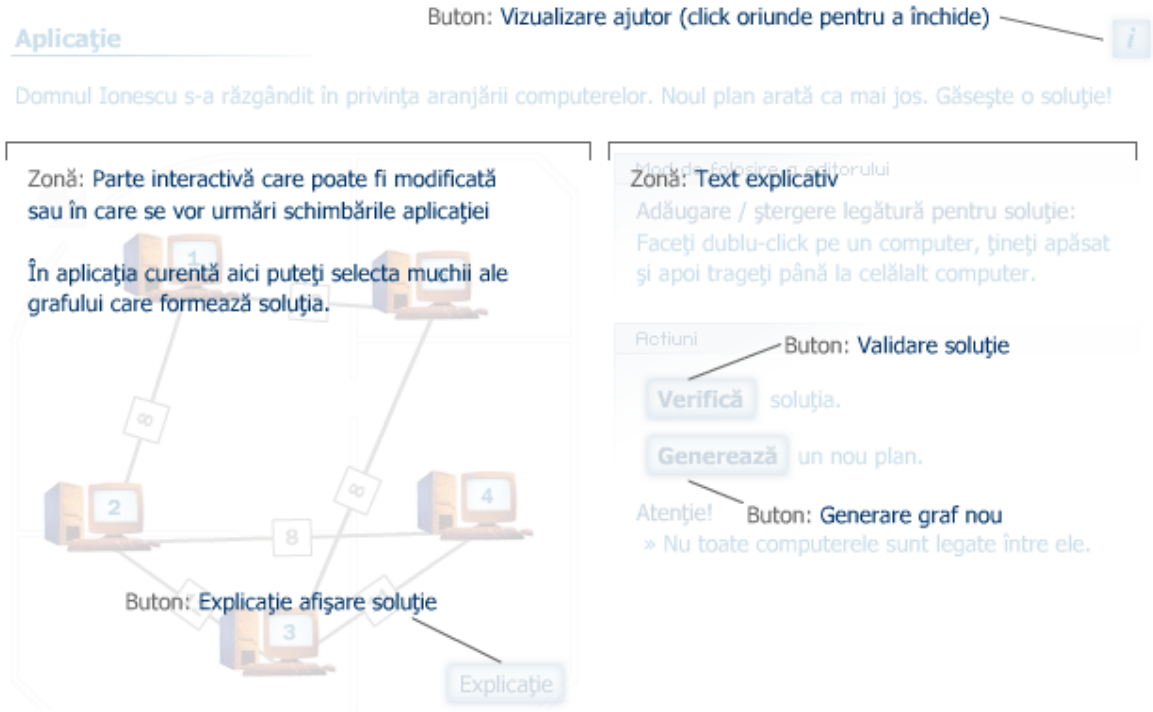
Butoane pentru închis ferestre detaliu –  – sunt amplasate în colțul dreapta sus a ferestrelor detaliu iar acționarea lor duce la închiderea ferestrei.

Buton de informații –  – la acționarea butonului se produce afişarea “deasupra” zonei de lucru a explicațiilor despre funcționalitatea fiecărui element de pe ecran.

Ferestre detaliu – sunt ferestre care oferă informații suplimentare despre o anumită noțiune. Asupra unei ferestre detaliu se poate face "*drag and drop*" acționând asupra oricărei zone a ferestrei. Exemplu :

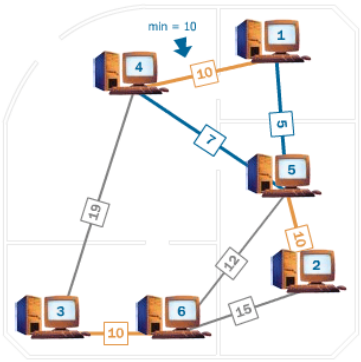


Ferestre de descriere a zonelor – sunt ferestre care la acționarea butonului  apar “deasupra” zonei de lucru şi conțin explicații despre funcționalitatea fiecărui element de pe ecran. Exemplu :



Ferestre eroare/atenționare – sunt ferestre detaliu care oferă informații despre unele erori făcute de utilizator sau atenționări în cazul nerezolvării tuturor problemelor.

Buton de vizualizare muchie care formează ciclu – Vizualizare – la acționare pune în evidență muchia de cost minim care, dacă ar fi selectată, ar forma un ciclu. Exemplu:



Buton vizualizare analiză complexitate – Complexitate – este un buton care, la acționare, deschide o fereastră detaliu în care sunt informații cu privire la ordinul de complexitate al algoritmului descris.

Buton vizualizare analiză complexitate – Corectitudine – este un buton care, la acționare, deschide o fereastră detaliu în care este demonstrată corectitudinea algoritmului.

Ferestre afişare sarcini de lucru – sunt ferestre în care sunt indicate anumite sarcini de lucru pentru momentul respectiv.

Ferestre afişare valori variabile – sunt ferestre în care sunt indicate valorile pe care le iau variabilele de memorie în timpul execuției programului (*watch*).

```
n: 4      m: 6      i: 1      NrMSel: 1
c: 1, 2, 3, 4
A: 1
G: { 4, 1, 10 }, { 1, 3, 11 }, { 2, 1, 11 },
   { 2, 4, 13 }, { 4, 3, 14 }, { 2, 3, 14 }
```

Ferestre afişare cod sursă – sunt ferestre în care este vizualizat codul sursă pe parcursul executării acestuia pas cu pasm punându-se în evidență instrucțiunea executată.

```
// unific componentele conexe
// ale extremitatilor muchiei i
if (c[G[i].e1]<c[G[i].e2])
{ min=c[G[i].e1];
 max=c[G[i].e2];
}
else
{ min=c[G[i].e2];
 max=c[G[i].e1];
```

2. Structura generală

În acest capitol sunt prezentate obiectivele didactice care pot fi atinse utilizând acest material. În finalul prezentării sunt incluse câteva recomandări privind unele moduri în care ar putea fi combinate aceste momente pentru a obține o lecție.

2.1. Obiective didactice

Obiectiv	Detaliere
Obiective de referință	
R1	Analizarea modului de determinare a arborilor parțiali de cost minim
R2	Realizarea aplicațiilor utilizând algoritmi specifici
R3	Urmărirea etapelor de realizare a unei aplicații
Obiective operaționale	
OP1	Definirea corectă a noțiunilor de graf neorientat, noduri, muchii
OP2	Definirea corectă a noțiunilor de graf conex, arbore, arbore parțial
OP3	Identificarea şi descrierea unor situații practice care să necesite determinarea unui arbore parțial de cost minim (APM)
OP4	Modelarea cu ajutorul teoriei grafurilor a situației practice prezentate
OP5	Formularea problemei practice în limbaj de grafuri
OP6	Definirea corectă a noțiunilor care intervin în formularea problemei în limbaj de grafuri
OP7	Formularea unei condiții necesare şi suficiente pentru ca un graf dat să admită arbori parțiali
OP8	Identificarea unui APM al unui graf conex specificat sau creat on-line
OP9	Alegerea reprezentării adecvate a grafului în memoria calculatorului
OP10	Descrierea structurilor de date necesare implementării algoritmului
OP11	Descrierea algoritmului lui Kruskal de determinare a unui APM
OP12	Implementarea algoritmului lui Kruskal într-un limbaj de programare
OP13	Urmărirea execuției “pas cu pas” a algoritmului lui Kruskal pe un exemplu specificat sau creat
OP14	Urmărirea valorilor variabilelor care intervin în desfăşurarea algoritmului lui Kruskal pe un exemplu
OP15	Descrierea algoritmului lui Prim de determinare a unui APM
OP16	Implementarea algoritmului lui Prim într-un limbaj de programare
OP17	Urmărirea execuției “pas cu pas” a algoritmului lui Prim pe un exemplu specificat sau creat
OP18	Urmărirea valorilor variabilelor care intervin în desfăşurarea algoritmului lui Prim pe un exemplu
OP19	Determinarea complexității timp a unui algoritm
OP20	Demonstrarea corectitudinii unui algoritm
OP21	Dezvoltarea gândirii algoritmice, logice, flexibile, creatoare
OP22	Dezvoltarea atenției concentrate şi spiritului de observație

2.2 Conținut

Se prezintă lista obiectelor de conținut (notate cu M) şi caracteristicile lor generale.

$M_{1.1}$ – Situație practică - prezentarea problemei	
Obiective didactice	OP1, OP2, OP3, OP21, OP22
Timp de predare	10 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea problemei • exemplificarea problemei pe un plan sau pe o hartă – predefinite • urmărirea determinării drumului minim de la punctul de start la toate celelalte puncte de pe hartă
Cuvinte cheie	• plan, hartă • graf neorientat • muchie • cost • cost minim

$M_{1.2}$ – Situație practică – aplicație1	
Obiective didactice	OP1, OP2, OP3, OP8, OP21, OP22
Timp de predare	15 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea unei situații concrete • prezentarea editorului • utilizarea editorului • determinarea unor soluții corecte şi explicarea lor
Cuvinte cheie	• plan, hartă • muchie • cost • cost minim

$M_{1.3}$ – Situație practică – aplicație2	
Obiective didactice	OP1, OP2, OP3, OP8, OP21, OP22
Timp de predare	15 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea unei situații concrete • prezentarea editorului • utilizarea editorului • determinarea unor soluții corecte şi explicarea lor
Cuvinte cheie	• plan, hartă • muchie • cost • cost minim

$M_{2.1}$ – Problema în limbaj de grafuri – asocierea unui graf	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP21, OP22
Timp de predare	20 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, algoritmizare, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea problemei • exemplificarea problemei pe o hartă predefinită • asocierea elementelor grafului element cu element
Cuvinte cheie	• plan, hartă • graf asociat • graf neorientat • nod • muchie • cost • cost minim

$M_{2.2}$ – Problema în limbaj de grafuri – prezentarea în limbaj de grafuri	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP21, OP22
Timp de predare	20 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• recapitularea noțiunii de graf conex • recapitularea noțiunii de arbore • prezentarea noțiunii de arbore parțial • prezentarea noțiunii de cost al arborelui parțial • prezentarea arborelui parțial de cost minim (APM) • verificarea APM pe două exemple
Cuvinte cheie	• graf asociat • graf conex • arbore • arbore parțial • costul muchiei • costul arborelui parțial • arbore parțial de cost minim (APM)

$M_{2.3}$ – Problema în limbaj de grafuri – evaluare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP7, OP8, OP21, OP22
Timp de predare	20 min
Tip de interacțiune cu elevii	• evaluare în formă scrisă prin intermediul calculatorului
Descriere	• exerciții de consolidare a cunoştințelor şi deprinderilor dobândite
Cuvinte cheie	• graf • cost minim • APM

$M_{3.1}$ – Algoritmul lui Kruskal – prezentare	
Obiective didactice	OP1, OP2, OP3, OP4, OP7, OP8, OP10, OP11, OP21, OP22
Timp de predare	25 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului în limbaj natural • prezentarea algoritmului pe paşi cu vizualizare pe un exemplu • prezentarea la fiecare pas a arborilor nou obținuți
Cuvinte cheie	• arbore, pădure • muchie, ciclu

$M_{3.2}$ – Algoritmul lui Kruskal – aplicație pseudocod	
Obiective didactice	OP1, OP2, OP3, OP4, OP7, OP8, OP10, OP11, OP21, OP22
Timp de predare	15 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului în limbaj natural • prezentarea algoritmului pe paşi cu vizualizare pe un exemplu • prezentarea la fiecare pas a arborilor nou obținuți
Cuvinte cheie	• arbore, pădure • muchie, ciclu

$M_{3.3}$ – Algoritmul lui Kruskal – evaluare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP7, OP8, OP21, OP22
Timp de lucru	20 min
Tip de interacțiune cu elevii	• evaluare în formă scrisă prin intermediul calculatorului
Descriere	• exerciții de verificare a cunoştințelor şi deprinderilor dobândite
Cuvinte cheie	• graf, graf parțial • cost minim • APM

$M_{4.1}$ – Algoritmul lui Kruskal – implementare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP9, OP10, OP11, OP12, OP13, OP14, OP19, OP21, OP22
Timp de predare	25 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului în limbaj natural • reprezentarea informațiilor necesare şi vizualizarea lor în codul sursă • identificarea unei muchii care nu formează un ciclu • vizualizarea fiecărui pas pe graful exemplu
Cuvinte cheie	• componentă conexă • complexitate • APM

$M_{4.2}$ – Algoritmul lui Kruskal – simulare implementare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP9, OP10, OP11, OP12, OP13, OP14, OP19, OP21, OP22
Timp de predare	25 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului pas cu pas • vizualizarea cosului sursă în timpul rulării • vizualizarea variabilelor de memorie şi a conținutului lor în timpul rulării programului • comentarea fiecărui pas al programului
Cuvinte cheie	• componentă conexă • complexitate • corectitudine • APM

$M_{5.1}$ – Algoritmul lui Prim – prezentare	
Obiective didactice	OP1, OP2, OP3, OP4, OP7, OP8, OP10, OP15, OP21, OP22
Timp de predare	25 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului în limbaj natural • prezentarea algoritmului pe paşi cu vizualizare pe un exemplu • prezentarea la fiecare pas a arborilor nou obținuți
Cuvinte cheie	• arbore, pădure • muchie, ciclu

$M_{5.2}$ – Algoritmul lui Prim – aplicație pseudocod	
Obiective didactice	OP1, OP2, OP3, OP4, OP7, OP8, OP10, OP15, OP21, OP22
Timp de predare	15 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului în limbaj natural • prezentarea algoritmului pe paşi cu vizualizare pe un exemplu • prezentarea la fiecare pas a arborilor nou obținuți
Cuvinte cheie	• arbore, pădure • muchie, ciclu

$M_{5.3}$ – Algoritmul lui Prim – evaluare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP7, OP8, OP21, OP22
Timp de lucru	20 min
Tip de interacțiune cu elevii	• evaluare în formă scrisă prin intermediul calculatorului
Descriere	• exerciții de verificare a cunoştințelor şi deprinderilor dobândite
Cuvinte cheie	• graf, graf parțial • cost minim • APM

$M_{6.1}$ – Algoritmul lui Prim – implementare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP9, OP10, OP15, OP16, OP17, OP18, OP19, OP21, OP22
Timp de predare	25 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului în limbaj natural • reprezentarea informațiilor necesare şi vizualizarea lor în codul sursă • identificarea unei muchii care nu formează un ciclu • vizualizarea fiecărui pas pe graful exemplu
Cuvinte cheie	• componentă conexă • complexitate • APM

$M_{6.2}$ – Algoritmul lui Prim – simulare implementare	
Obiective didactice	OP1, OP2, OP3, OP4, OP5, OP6, OP9, OP10, OP15, OP16, OP17, OP18, OP19, OP21, OP22
Timp de predare	25 min
Tip de interacțiune cu elevii	• **metode de comunicare orală**: expunere, conversație, studiu de caz • **metode de acțiune**: exercițiul, învățarea prin descoperire • **procedee de instruire**: explicația în etapa de comunicare; învățarea prin descoperire dirijată, inductivă, experimentală, exercițiul de consolidare
Descriere	• prezentarea algoritmului pas cu pas • vizualizarea cosului sursă în timpul rulării • vizualizarea variabilelor de memorie şi a conținutului lor în timpul rulării programului • comentarea fiecărui pas al programului
Cuvinte cheie	• componentă conexă • complexitate • corectitudine • APM

2.3. Recomandări de structurare şi predare

- **Planul unității de învățare 1** **Timp: 1 oră**

Obiect de conținut	Timp (min)
$M_{1,1}$	10
$M_{1,2}$	10
$M_{1,3}$	10
$M_{2,1}$	20

- **Planul unității de învățare 2** **Timp: 1 oră**

Obiect de conținut	Timp (min)
$M_{2,2}$	30
$M_{2,3}$	20

- **Planul unității de învățare 3** **Timp: 1 oră**

Obiect de conținut	Timp (min)
$M_{3,1}$	20
$M_{3,2}$	10
$M_{3,3}$	20

- **Planul unității de învățare 4** **Timp: 1 oră**

Obiect de conținut	Timp (min)
$M_{4,1}$	25
$M_{4,2}$	25

- **Planul unității de învățare 5** **Timp: 1 oră**

Obiect de conținut	Timp (min)
$M_{5,1}$	20
$M_{5,2}$	10
$M_{5,3}$	20

- **Planul unității de învățare 6** **Timp: 1 oră**

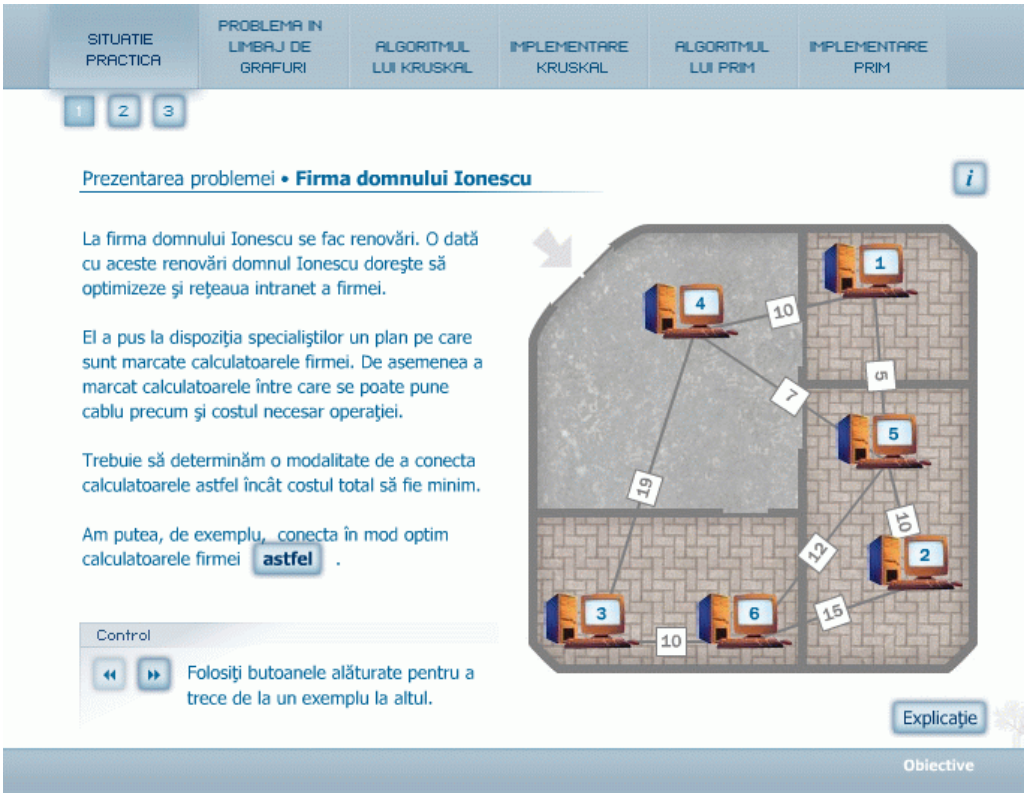
Obiect de conținut	Timp (min)
$M_{6,1}$	25
$M_{6,2}$	25

3. Obiecte de conținut - detaliere

În continuare vom prezenta în detaliu modul de utilizare a elementelor din ferestrele lecției (navigare, elemente specifice, funcționarea aplicațiilor, etc.). Subliniem că navigarea elementară se face cu ajutorul butoanelor descrise în Capitolul 1 – Terminologie, al acestui manual. Nu ne vom referi la acestea decât spicuitiv.

3.1. Situația practică – prezentarea problemei

În acest obiect de conținut este prezentată o problemă practică care necesită utilizarea unui algoritm de determinare a unui arbore parțial de cost minim. Tot aici este posibilă descrierea acestei situații pe o hartă sau pe un plan şi exemplificarea determinarii unui arbore de cost minim pe 2 situații practice date.



Ecranul este împărțit în patru zone :

- în partea de sus zona meniurilor care descriu obiectele de conținut şi momentele acestora
- bara de jos conține totdeauna un singur buton Obiective
- în dreapta zona unde apare situația practică
- în stâga acesteia zona text unde este descrisă situația practică

Pe lângă aceste zone în fereastră mai apar câteva butoane şi controale:


butoane control – cele două butoane au efectul de a produce trecerea înainte / înapoi de la o situație practică la alta

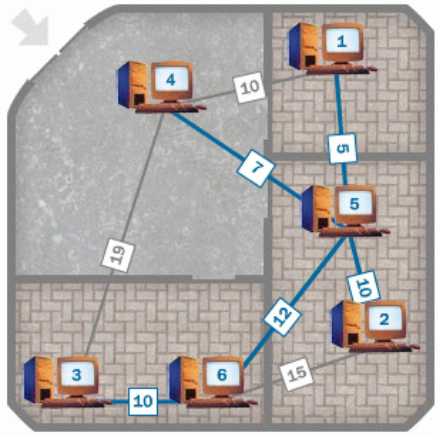

buton informații – care acționat permite vizualizarea obiectelor din fereastră şi rolul lor


la acționarea acestui buton în zona din dreapta, pe plan/hartă apare figurată soluția problemei practice

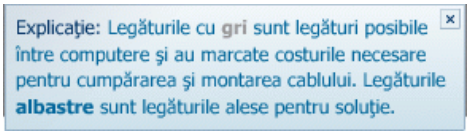

la acționarea acestui buton într-o fereastă detaliu apar explicații pentru soluția dată

Obiectul de conținut cuprinde descrierea problemei practice în limbaj natural, precum şi sarcinile de lucru. Tot aici sunt date informații despre zonele ecranului. Aceste informații se obțin acționând butonul  . La acționarea lui "deasupra" ecranului apare descrierea zonelor şi butoanelor ecranului, acesta rămânând vizibil în spate.

După citirea situației practice, butonul  oferă posibilitatea vizualizării soluției prin marcarea cu linii albastre a conectării corecte a calculatoarelor:



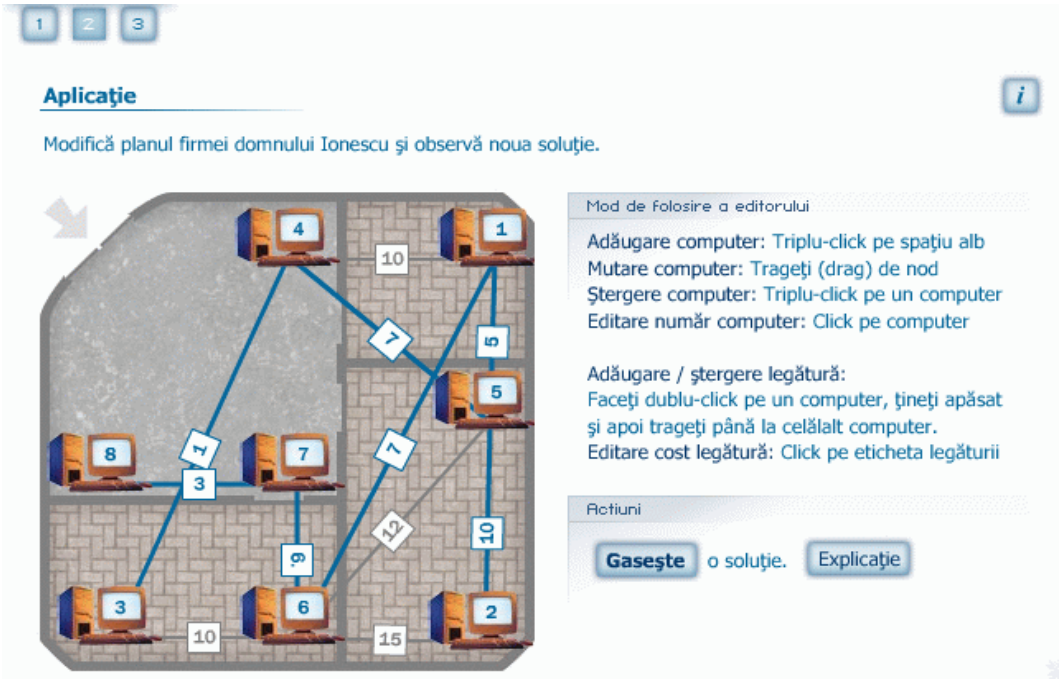
iar acționarea butonului  permite vizualizarea explicațiilor:



Trecea la o altă situație practică se face utilizând butoanele de control.

3.2. Situația practică – aplicație1

În acest obiect de conținut elevului i se dă ca sarcină de lucru să-şi construiască singur un plan de conectare a calculatoarelor sau o nouă hartă a oraşelor, să modifice numărul de calculatoare / oraşe, să editeze numerotarea acestora şi să modifice lungimile cablurilor / drumurilor care leagă calculatoarele / oraşele.



Pentru a realiza această sarcină i se pune la dispoziție elevului un editor a cărui mod de utilizare este descris în partea din dreapta a ecranului. Operațiile permise:

- adăugare / ştergere computer / oraş – triplu click
- adăugare / ştergere legătură / drum – dublu click apoi drag
- mutare computer / oraş – drag&drop
- editare număr computer / oraş sau cost legătură / drum – click pe etichetă, apoi editare de la tastatură; finalizare cu ENTER

În orice moment la acționarea butonului Găseşte se generează o soluție vizualizată pe hartă prin modificarea culorii legăturilor, aşa cum i se explică elevului prin acționarea butonului Explicaţie.

3.3. Situația practică – aplicație2

În acest obiect de conținut elevului i se cere să găsească singur o soluție pentru legarea computerelor / reabilitarea drumurilor. Fereastra principală este organizată astfel:

În partea stângă a ferestrei de lucru se găseşte un nou plan de aranjare a computerelor / oraşelor. Sarcina de lucru apare scrisă imediat deasupra.

Pentru a putea lucra elevul are la dispoziție doar una dintre facilitățile editorului, şi anume aceea care îi permite să adauge / şteargă o legătură. Acest lucru este indicat în partea din dreapta sus a ferestrei de lucru.

După adăugarea legăturilor, la acționarea butonului Verifică soluția oferită de elev este verificată, şi, în cazul în care este corectă, acest lucru este semnalat printr-un mesaj: "Felicitări! Ai obținut o soluție validă." Dacă însă soluția nu este corectă sau nu au fost stabilite legături între toate computerele / oraşele, elevul va fi atenționat: "Atenție! Nu toate computerele sunt legate între ele.", respectiv "Atenție! Cost prea mare: 47".

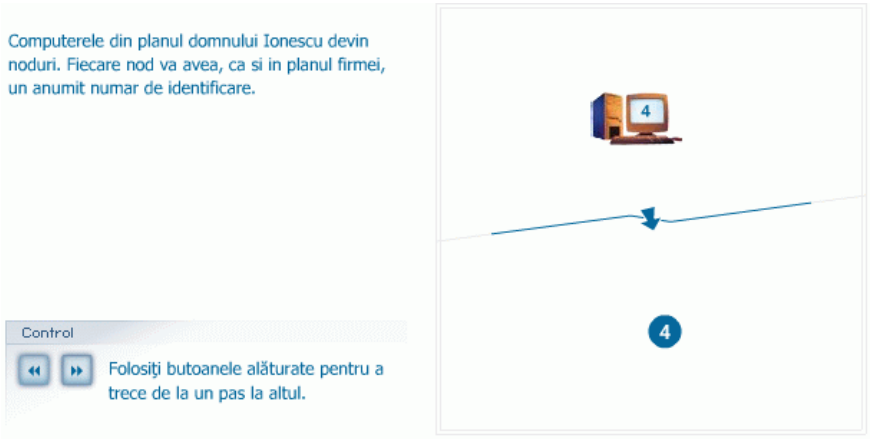
În orice moment există posibilitatea generării unui nou plan / hartă prin acționarea butonului Generează.

3.4. Problema în limbaj de grafuri – asocierea unui graf problemei practice

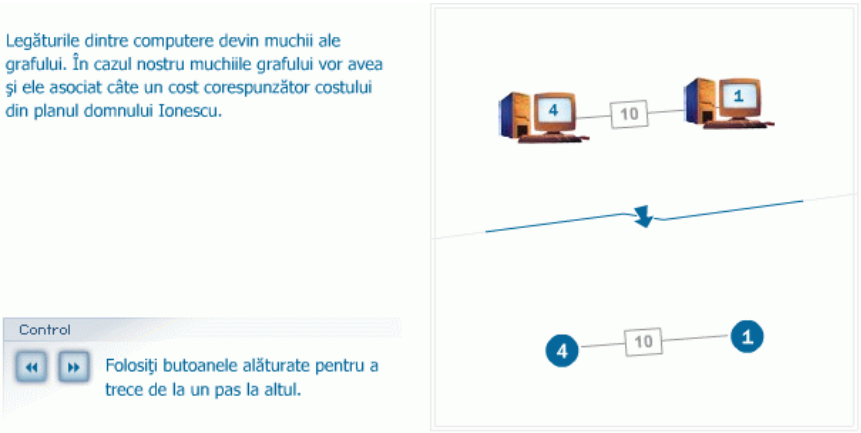
În acest obiect de conținut este prezentată modalitatea în care se poate asocia problemei practice, un graf. Acest lucru este realizat pas cu pas, conducând elevul spre înțelegerea acestui fenomen.

Trecerea de la un moment la altul este realizată cu ajutorul butoanelor de control 

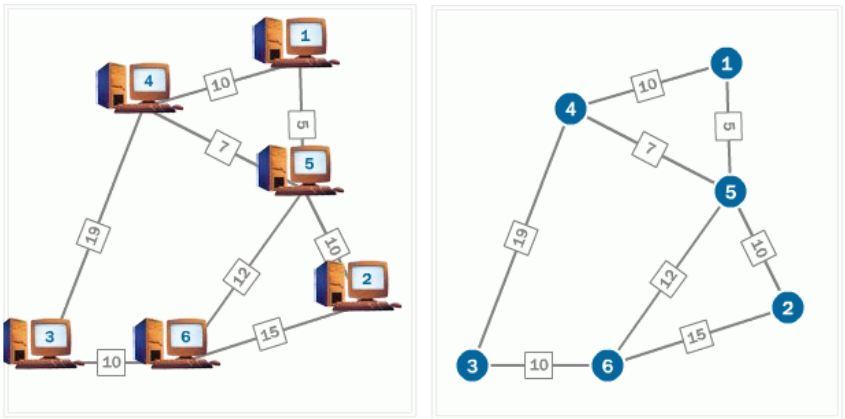
La primul pas fiecărui computer / oraş i se asociază un nod:



La pasul următor legăturilor dintre computere li se asociază câte o muchie în graf:

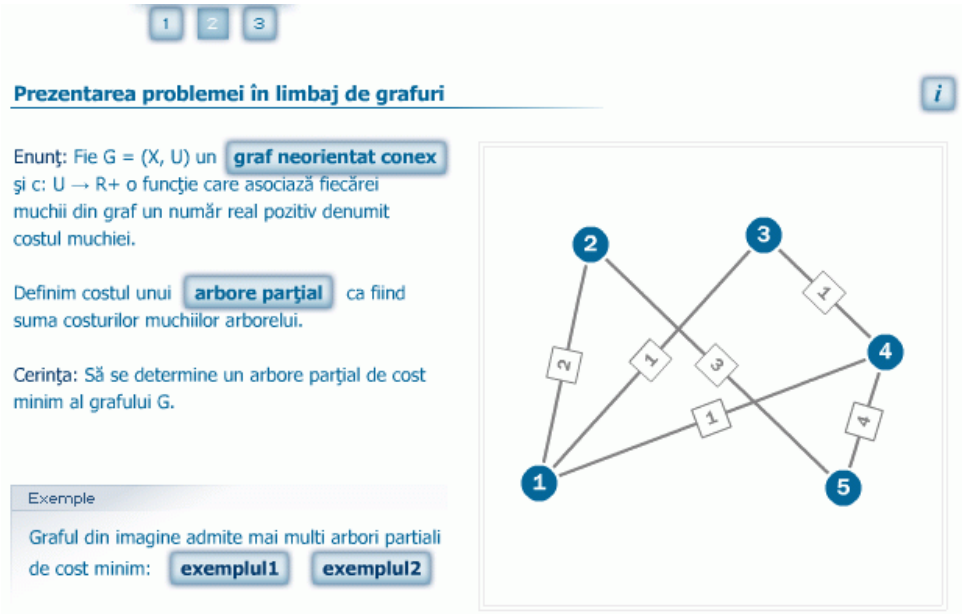


La final, întregul plan este reprezentat printr-un graf:



3.5. Problema în limbaj de grafuri – prezentarea problemei în limbaj de grafuri

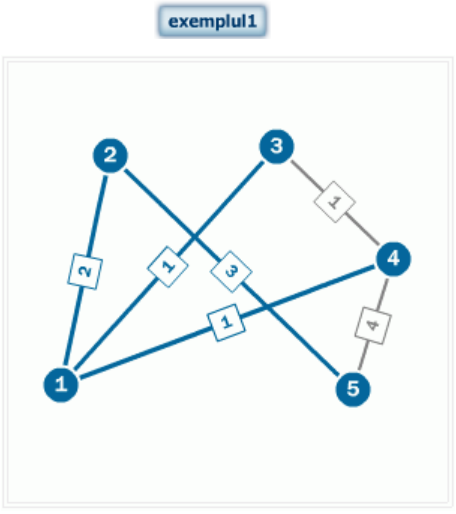
În acest obiect de conținut elevul învață cum problema practică devine o problemă de grafuri.

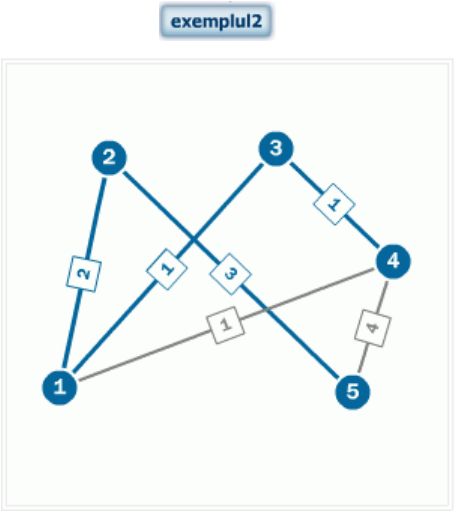


Fereastra de lucru conține enunțarea problemei în limbaj de grafuri, precum şi posibilitatea de a revedea definițiile unor elemente din teoria grafurilor necesare înțelegerii problemei (butoanele graf neorientat conex arbore parţial).

Se precizează apoi că APM nu este neapărat unic, pentru exemplul dat existând doi arbori parțiali de cost minim, care pot fi vizualizați acționând butoanele

sau





3.6. Problema în limbaj de grafuri – evaluare

Acest obiect de conținut constă dintr-un set de două exerciții recapitulative care se selectează cu ajutorul butoanelor de forma 1 .

Exercițiul 1, este de tip test grilă cu răspunsuri de tip “complement simplu”, adică doar o variantă de răspuns corectă.

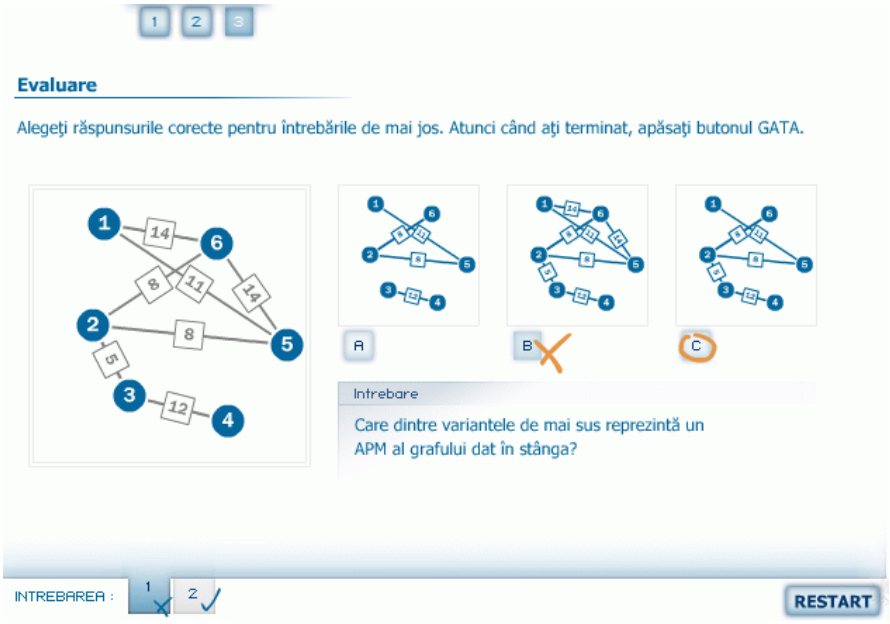
Exercițiul 2 este de tip interactiv, solicitând construirea unui APM pe un graf dat utilizând facilitățile oferite de editorul implementat.

- exercițiul 1 cere selectarea unui APM dintre cele trei variante oferite; selecția se face utilizând butoanele de selecție de tipul B
- exercițiul 2 costruirea unui APM utilizând mouse-ul.

Exercițiile se selectează cu ajutorul mouse-ului.

Finalizarea evaluării se face prin acționarea butonului GATA . La acționarea acestuia, se verifică în primul rând dacă elevul a răspuns sau nu la ambele întrebări. Dacă nu a răspuns apare un mesaj de atenționare.

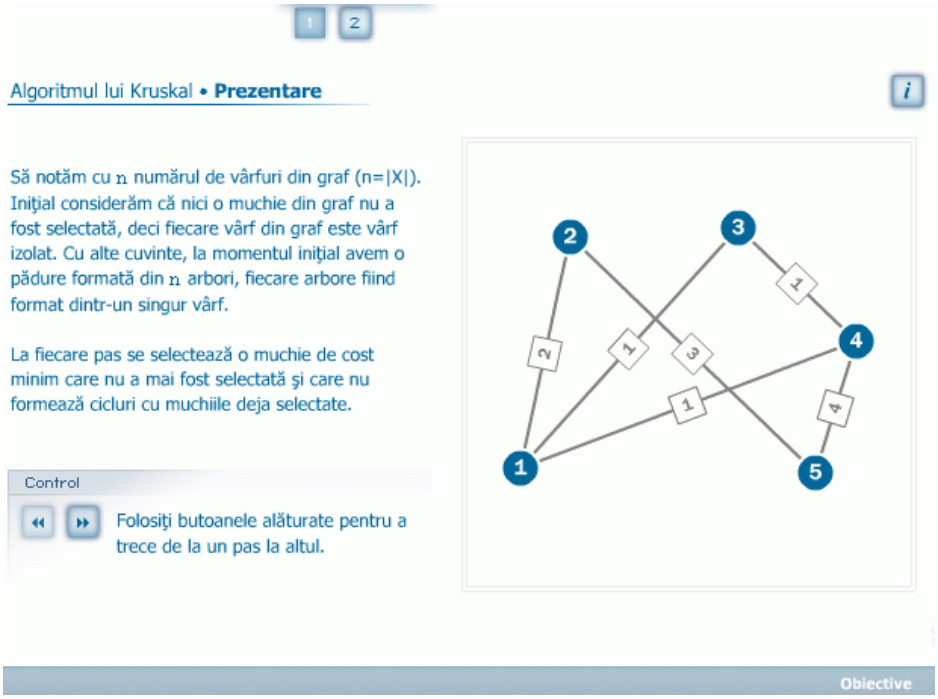
Dacă s-a răspuns la ambele întrebări, acestea sunt validate. Dacă răpspunsul a fost corect apare pe buton semnul 2 însoțit de un mesaj de felicitare, în caz contrar răspunsul este invalidat 1 , indicându-se şi soluția corectă:



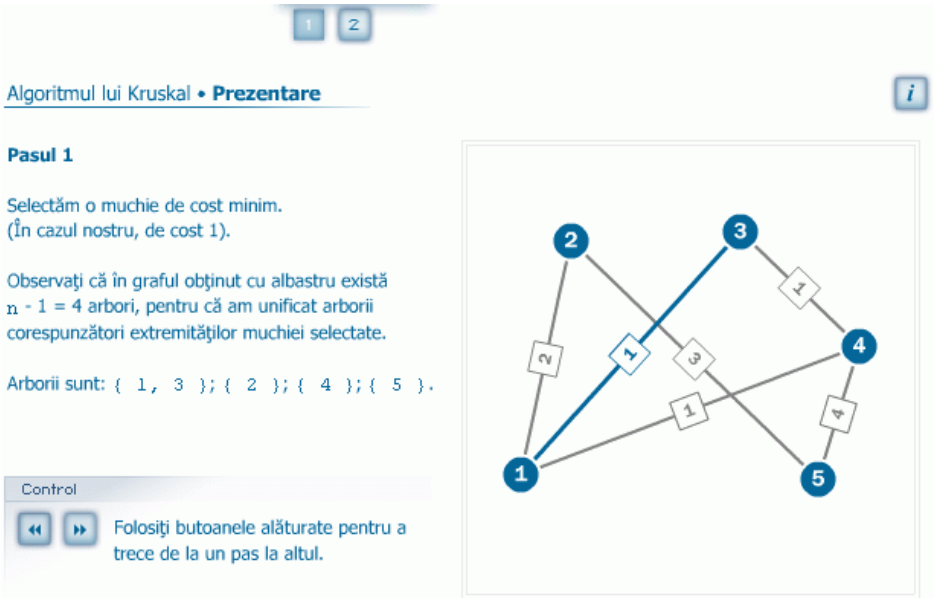
Efectuarea unor noi exerciții se poate face acționând butonul RESTART .

3.7. Algoritmul lui Kruskal – prezentare

Acest obiect de conținut constă din prezentarea algoritmului în limbaj natural şi din vizualizarea însoțită de explicații a fiecărui pas din descriere.



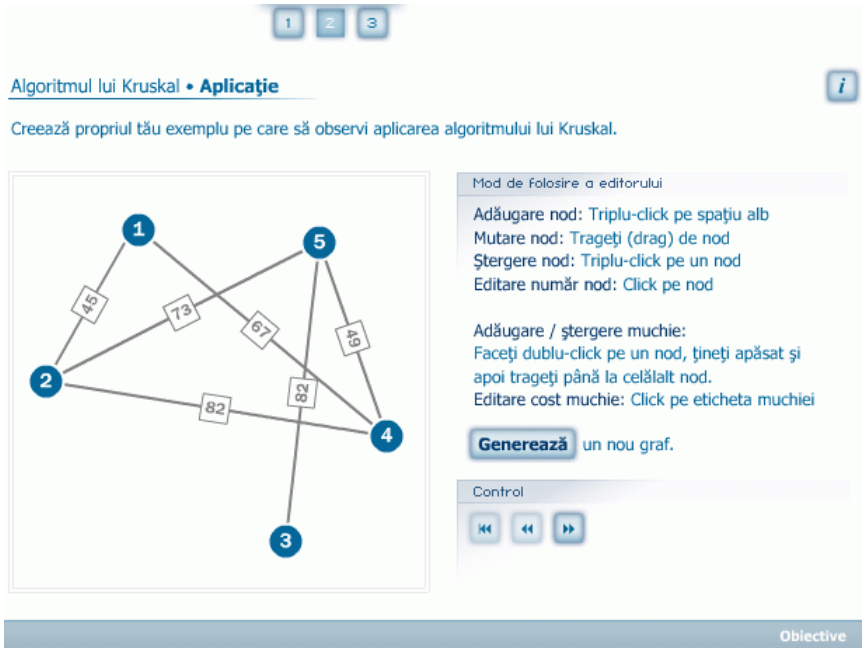
Trecerea de la un pas la altul se face utilizând butoanele de control  . La fiecare pas se comentează în partea din stânga a ferestrei de lucru în limbaj natural pasul care se execută în desfăşurarea algoritmului, indicându-se operația efectuată şi rezultatul acesteia, iar în partea dreaptă a ferestrei se vizualizează pe graful exemplu acest lucru. Exemplu, pasul 1:



În final, după patru paşi se obține un APM.

3.8. Algoritmul lui Kruskal – aplicație

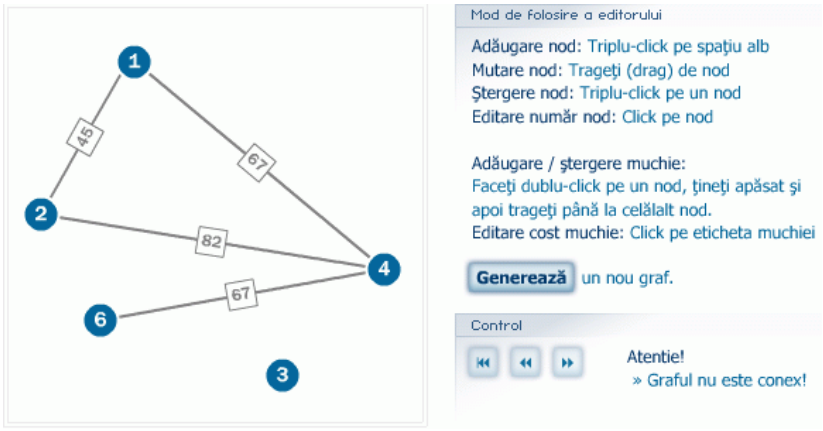
În acest obiect de conținut elevul are posibilitatea să-şi creeze propriul exemplu pe care să poată urmări pas cu pas desfăşurarea algoritmului descris în momentul de conținut 3.7.



Pentru a realiza acest lucru i se pune la dispoziție elevului un graf conex generat aleator şi un editor de grafuri care permite operațiile:

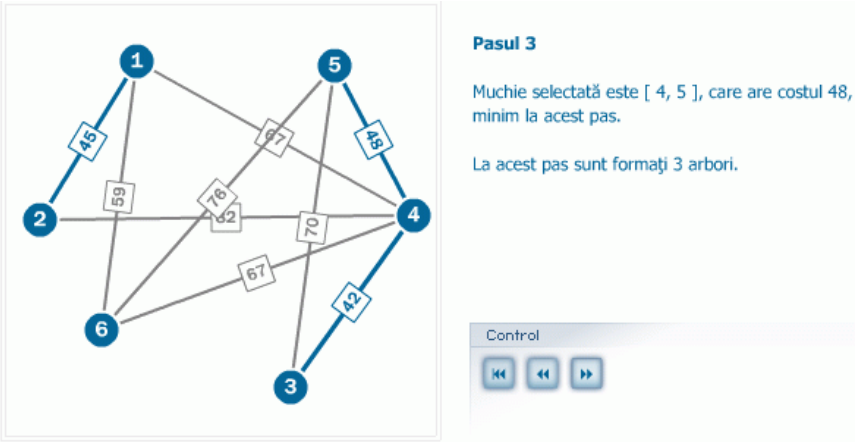
- adăugare / ştergere nod – triplu click
- adăugare / ştergere muchie – dublu click pe un nod apoi drag până la alt nod
- mutare nod – drag&drop
- editare etichetă nod sau cost muchie – click pe etichetă / cost, apoi se introduc noile valori de la tastatură; operația se încheie cu ENTER

În cazul în care, în timpul modificării grafului, acesta nu mai îndeplineşte condiția de conexitate, un mesaj avertizează asupra acesti lucru şi butoanele controlului pentru derularea aplicației sunt blocate până la restabilirea condiției de conexitate:



După restabilirea proprietății de conexitate butonul devine activ şi aplicația se poate desfăşura. În acelaşi timp este disponibil butonul Generează care permite generarea unui nou graf conex.

După ce aplicația este lansată în execuție, la fiecare pas este indicată muchia care s-a selectat, costul acesteia precum şi numărul de arbori care au mai rămas.



În timpul derulării aplicației, există disponibil în orice moment butonul  care realizează părăsirea aplicației în curs de desfăşurare şi generarea imediată a unui nou graf conex, deci reluarea aplicației pe un alt exemplu.

În timp ce controlul este activ nu mai este permisă modificarea nici unei caracteristici a grafului pe care se desfăşoară aplicația.

3.9. Algoritmul lui Kruskal – evaluare

Acest obiect de conținut constă dintr-un set de trei exerciții recapitulative care se selectează cu ajutorul butoanelor de forma 1 .

Exercițiul 1 este de tip interactiv, solicitând construirea unui APM pe un graf dat utilizând facilitățile oferite de editorul implementat.

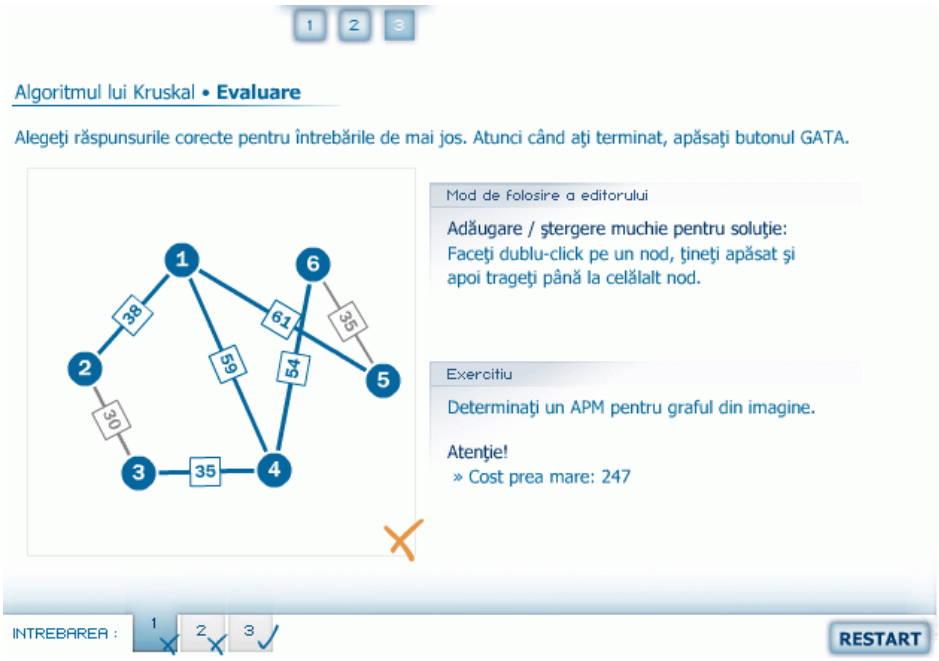
Exercițiile 2 şi 3 sunt de tip test grilă cu răspunsuri de tip “complement simplu”, adică doar o variantă de răspuns corectă.

- exercițiul 1 cere costruirea unui APM utilizând mouse-ul şi facilitățile editorului
- exercițiul 2 cere determinarea următoarei muchii selectate în desfăşurarea algoritmului lui Kruskal, dintre cele trei variante oferite; selecția se face utilizând butoanele de selecție de tipul B
- exercițiul 3 cere determinarea arborelui parțial obținut la un momemnt dat în desfăşurarea algoritmului lui Kruskal, dintre cele trei variante oferite; selecția se face utilizând butoanele de selecție de tipul B

Exercițiile se selectează cu ajutorul mouse-ului.

Finalizarea evaluării se face prin acționarea butonului GATA . La acționarea acestuia, se verifică în primul rând dacă elevul a răspuns sau nu la toate întrebările. Dacă nu a răspuns apare un mesaj de atenționare.

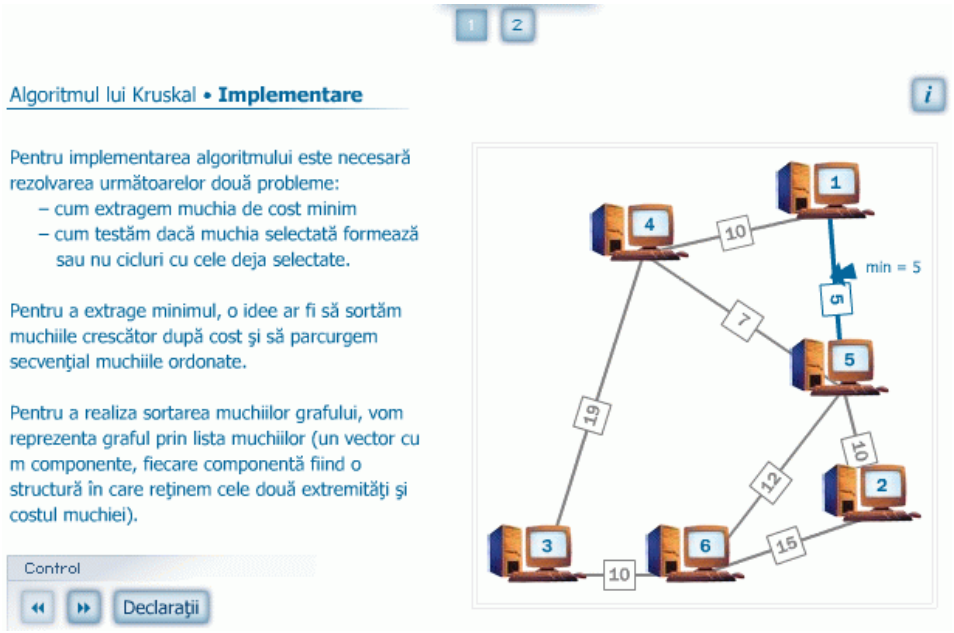
Dacă s-a răspuns la toate cele trei întrebări, acestea sunt validate. Dacă răspunsul a fost corect apare pe buton semnul 2 însoțit de un mesaj de felicitare, în caz contrar răspunsul este invalidat 1 , indicându-se şi eventuala eroare:



Efectuarea unor noi exerciții se poate face acționând butonul RESTART .

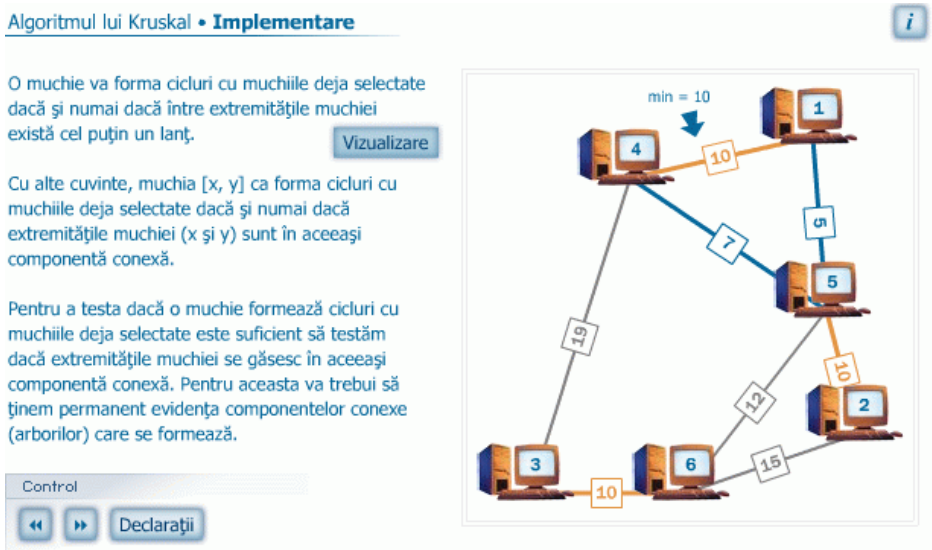
3.10. Algoritmul lui Kruskal – implementare

În acest obiect de conținut sunt prezentate noțiunile necesare implementării algoritmului: modalitatea în care sunt selectate muchiile care vor forma APM, datele implicate, organizarea lor, semnificația notațiilor folosite.



Inițial, în fereastra de lucru, în partea dreaptă este figurat graful, cu prima muchie deja selectată, iar în partea stângă sunt indicate problemele care trebuie rezolvate pentru determinarea APM. Tot aici este indicată modalitatea de a reprezenta graful pentru a putea rezolva prima problemă – sortarea muchiilor crescător după cost.

În a doua secvență a obiectului de conținut (după prima acționare a butonului)este descrisă, în partea stângă, în limbaj natural, modalitatea de rezolvare a celei de a doua probleme care apare şi anume "determinarea muchiei de cost minim dintre cele rămase care să nu formeze un ciclu cu muchiile deja selectate". În acest moment devine activ şi butonul Vizualizare care indică pe graful din dreapta una dintre muchiile de cost minim care, dacă ar fi selectată, ar forma un ciclu cu două dintre muchiile deja selectate.



În următoarea secvență a obiectului de conținut (care se obține după a doua acționare a butonului) sunt indicate structurile de date necesare implementării algoritmului precum şi modul lor de reprezentare:

`n` – numărul de vârfuri

`m` – numărul de muchii din graf

`G` – graful dat, reprezentat prin lista muchiilor (un vector cu `m` componente, fiecare componentă fiind o structură în care reținem cele două extremități şi costul muchiei)

`A` – arborele parțial de cost mimim, reprezentat ca un vector `n-1` componente în care vom reține indicii din `G` ai muchiilor selectate

`c` – vector cu `n` componente în care vom reține evidența componentelor conexe (`c[i]` = componenta conexă careia îi aparține vârful `i`). Componentele conexe vor fi identificate printr-un reprezentant.

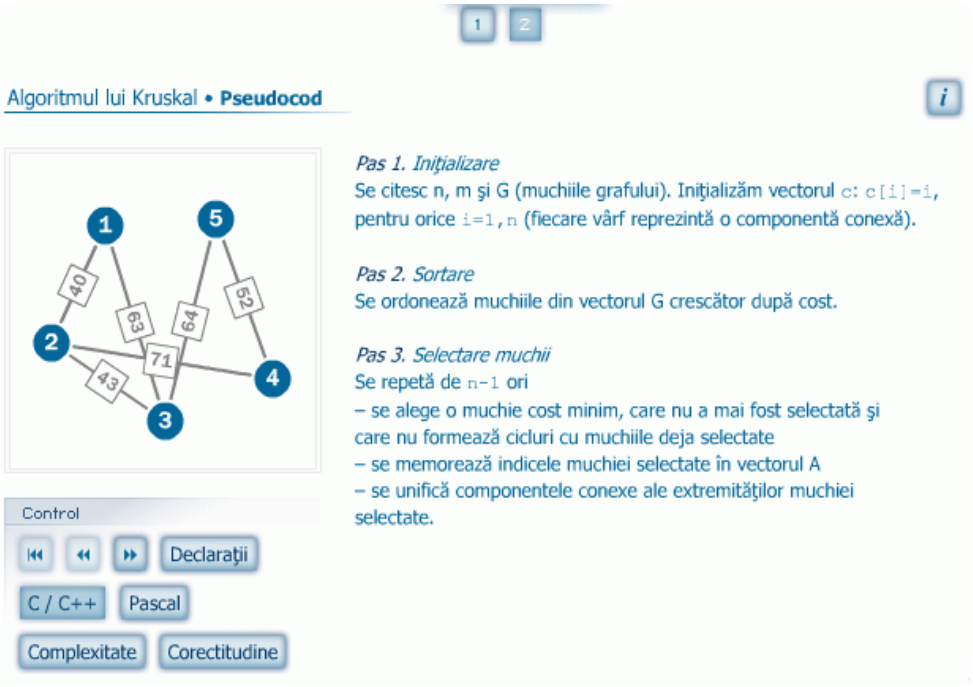
Pentru a vizualiza structurile de date utilizate, avem la dispoziție butonul Declarații, la acționarea căruia se vizualizează zona de declarații a variabilelor din codul sursă:

```
#define NMaxVf 50
#define NMaxMuchii NMaxVf*(NMaxVf-1)/2
struct Muchie {int e1, e2, cost; };
Muchie G[NMaxMuchii];
int A[NMaxVf], c[NMaxVf];
int n, m;
```

Utilizând butoanele de control se poate naviga prin cele trei secvențe ale obiectului de conținut până la completa înțelegere a fenomenului.

3.11. Algoritmul lui Kruskal – simulare implementare

Acest obiect de conținut prezintă implementarea algoritmului lui Kruskal de determinare a APM, atât sub formă de pseudocod, cât şi sub formă de cod sursă C++ şi cod sursă Pascal.



În prima secvență a obiectului de conținut, în partea dreaptă a ferestrei de lucru, este descris algoritmul în limbaj natural sintetizând elementele descrise în momentele anterioare. În partea stângă a ferestrei de lucru este dat un graf exemplu, iar dintre butoanele controlului este activ doar butonul ⏩. Toate celelate butoane sunt active, dar în acest moment doar butonul Declaraţii prezintă interes.

După acționarea butonului de control ⏩ se trece la **pasul 1**(Inițializare). În acest moment toate butoanele de control devin active, iar graful exemplu devine editabil. Pentru ca graful să poată fi modificat, elevului i se pune la dispoziție editorul încorporat aplicației şi, în acelaşi timp, în partea dreaptă a ferestrei de lucru se indică comenzile necesare pentru a putea face modificările necesare cu ajutorul mouse-ului.

Tot acum, dacă elevul doreşte să genereze un alt graf, poate să facă acest lucru prin intermediul butonului **Generează**.

O altă facilitate oferită de aplicație în acest moment apare în partea din dreapta jos a ferestrei de lucru şi constă în lista de variabile care apar în implementarea algoritmului lui Kruskal şi a valorilor pe care acestea le primesc în timpul rulării programului (corespunde ferestrei "Watch" din mediile de programare).

```
n: 4       m: 6       i: 1       NrMSel: 1
c: 1, 2, 3, 4
A: 1
G: { 4, 1, 10 }, { 1, 3, 11 }, { 2, 1, 11 },
   { 2, 4, 13 }, { 4, 3, 14 }, { 2, 3, 14 }
```

Fiecare muchie are inițial un anumit cost care apare în graful G reprezentat prin lista de adiacență. Dacă se fac modificări ale costurilor unor muchii, acestea vor fi imediat vizibile în lista G.

În cazul în care, în timpul modificării grafului, acesta nu mai îndeplineşte condiția de conexitate, butonul de control pentru derularea aplicației devine blocat până la restabilirea condiției de conexitate.

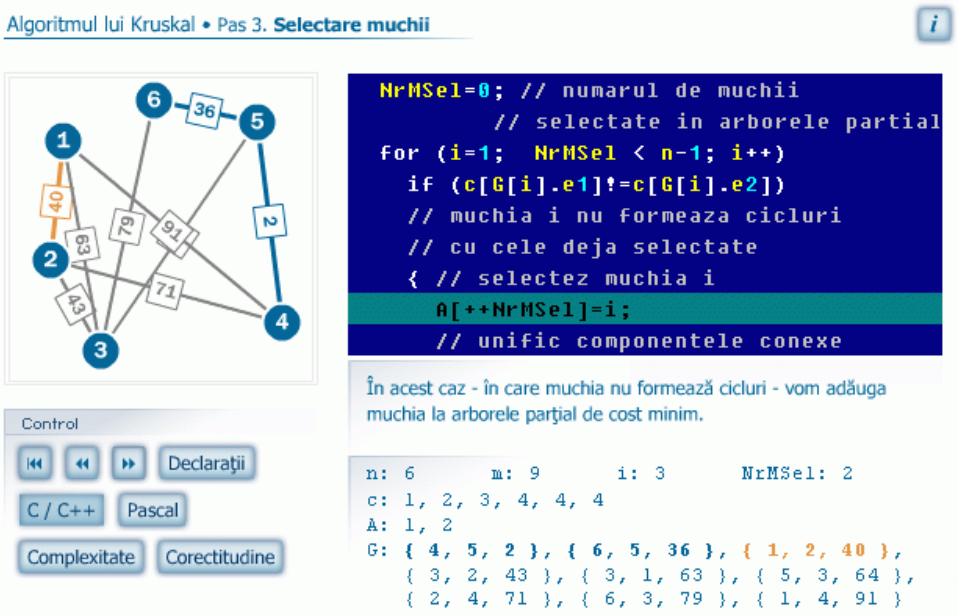
Acționarea butonului de control produce trecerea la **pasul 2** al algoritmului şi anume sortarea muchiilor din vectorul G după valorile crescătoare a valorilor costurilor. În partea din dreapta sus a ferestrei de lucru este vizualizat segmentul de cod care realizează sortarea:

```
void SortareMuchii(int st, int dr)
{int i, j; Muchie x;
 if (st<dr)
   { x=G[st]; i=st; j=dr;
     while (i<j)
     { while (i<j&&G[j].cost>=x.cost)
         j--;
       G[i]=G[j];
       while (i<j && G[i].cost<=x.cost)
```

În fereastră nu se vede întreg codul, dar făcând un clic în fereastră, aceasta devine activă şi, cu ajutorul săgeților de pe tastatură sus-jos, întreg codul poate fi vizualizat.

În partea din stânga a ferestrei de lucru, sub butoanele de control, există două butoane care permit comutarea vizualizării codului sursă între cele două limbaje cae se învață în liceu (C++ / Pascal) C / C++ Pascal.

La acționarea butonului de control se trece la **pasul 3** al algoritmului, care se va efectua în mod repetat prin acționarea aceluiaşi buton de control până la selectarea a `n-1` muchii.



Pe parcursul executării algoritmului în fereastra de cod se indică în fiecare moment zona de cod executată iar pe graf se figurează muchiile selectate şi cea care este analizată în acel moment. Sub zona de cod C++ / Pascal există o fereastră în care este comentat fiecare pas al algoritmului. În acelaşi timp, în zona de variabile sub vizibile toate modificările petrecute pe parcursul execuției codului sursă.

În timpul derulării aplicației, există disponibil în orice moment butonul care realizează părăsirea aplicației în curs de desfăşurare şi generarea imediată a unui nou graf conex, deci reluarea aplicației pe un alt exemplu.

Algoritmul fiind descris şi implementat, în acest moment pot fi acționate cele două butoane care descriu:

Complexitate – informații cu privire la ordinul de complexitate al algoritmului descris

Complexitatea algoritmului
Pas 1: Iniţializare
O(m) pentru citirea datelor şi O(n) iniţializarea vectorului c.
Pas 2: Sortare
Complexitatea acestui pas depinde de algoritmul de sortare folosit.
În implementarea acestui pas am utilizat algoritmul de sortare rapidă (QuickSort), deci complexitatea în medie este O(m log m).
Pas 3: Selecţie muchii
Pentru a selecta cele n-1 muchii ale arborelui parţial de cost minim se va parcurge (în cazul cel mai defavorabil) toată lista muchiilor.
Pentru fiecare muchie selectată se realizează unificarea componentelor conexe ale extremităţilor muchiei, operaţie de complexitate O(n) în implementarea prezentată.
Deci complexitatea la acest pas va fi O(m+n2).

Observaţii
1.În cazul în care arborele parţial de cost minim este găsit suficient de repede, un număr mare de muchii rămân netestate şi în acest caz se pierde timp inutil cu sortarea acestor muchii. O altă idee, mai eficientă, ar fi să organizăm muchiile grafului ca un min-heap, structură ce permite extragerea eficientă a muchiei de cost minim. În acest caz, pentru organizarea muchiilor grafului ca heap sunt necesare O(m) operaţii.
Pentru extragerea fiecarei muchii de cost minim sunt necesare O(m log m) operaţii.
2.Selectarea unei muchii implică şi o operaţie de unificare a două componente conexe, operaţie pe care am implementat-o în O(n). Acest pas se poate optimiza până la o complexitate medie de O(log n).

Corectitudine – demonstrația corectitudinii algoritmului

Teoremă de corectitudine
Algoritmul lui Kruskal generează un arbore parţial de cost minim.

Demonstraţie
1. Algoritmul selectează numărul maxim de muchii care nu formează cicluri, deci, conform teoremei de caracterizare a arborilor, se obţine un arbore parţial.
2. Să demonstrăm că arborele parţial obţinut în urma aplicării algoritmului lui Kruskal este un arbore parţial de cost minim:
Fie A = (a1, a2, ..., an-1) muchiile arborelui rezultat în urma aplicării algoritmului, în ordinea selectării lor.
Presupunem prin reducere la absurd că arborele obţinut nu este de cost minim, deci există
A' = (a1', a2', ..., an-1') un alt arbore parţial, astfel încât c(A') < c(A).
Fie k = min { i | 1 ≤ i ≤ n, ai ≠ ai' }, primul indice de la care A şi A' diferă.
Deci A = (a1, a2, ..., ai-1, ai, ..., an-1)
A' = (a1, a2, ..., ai-1, ai', .., an-1'), cu ai ≠ ai' .
Evident c(ai) ≤ c(aj'), j aparţine { i, ..., n-1 }, altfel algoritmul ar fi selectat muchia aj' în loc de ai, deoarece ar fi avut cost mai mic şi nu formează cicluri cu a1,...,ai-1. Adaug la A' muchia ai. Se formează un ciclu în care intervin doar muchii din {ai'...an-1'}. Elimin una din muchiile ciclului diferită de ai. Se obţine un arbore A" = (a1,..., ai-1, ai, ai+1"..., an-1") care are i muchii comune cu A. În plus c(A")-c(A') = c(ai) - c(aj') ≤ 0, deci c(A") ≤ c(A').
Repetăm procedeul de înlocuire a muchiilor din A' cu muchiile din A. Obţinem c(A') ≥ c(A") ≥ ... ≥ c(A). Dar am presupus că c(A') < c(A), deci contradicţie. Deci A este un arbore parţial de cost minim.

3.12. Algoritmul Prim – prezentare
3.13. Algoritmul Prim – aplicație
3.14. Algoritmul Prim – evaluare
3.15. Algoritmul Prim – implementare
3.16. Algoritmul Prim – simulare implementare

Pentru obiectele de conținut 3.12. – 3.16. elementele prezentate în obiectele de conținut 3.7. – 3.11. rămân valabile, algoritmul prezentat fiind de data aceasta algoritmul lui Prim.

4. Elemente de implementare a aplicației

Structuri de date utilizate

Fiecare dintre momente este creat ca un fiind un fişier Flash distinct, însă toate acestea folosesc module comune pentru a menține extensibilitatea aplicației. Deasemenea, anumite exemple de grafuri (în special acelea care nu sunt generate dinamic) sunt stocate în fişiere XML pentru o foarte mare portabilitate. Astfel, un graf poate fi editat în orice editor text, chiar Notepad.

Structura aplicației

Aplicația, în sistemul de fişiere este organizată astfel:

- content – director ce conține momentele lecției
- classes – director ce conține clasele modulelor folosite în lecție
- M*.fla, M*.swf – momentele lecțiilor, E*.xml - grafuri
- CBar.as, CMenu.as, CMenuButton.as, CSubMenu.as, CSubMenuButton.as – clase ale modulelor folosite doar în aplicația centrală
- menu-initialize.as – fişier ce conține structura meniului din aplicația centrală
- index.html, central.fla, central.swf – aplicația centrală
- central.flp, full.flp – fişiere-proiect Flash

Resurse Hardware şi Software

Aplicația poate rula pe aproape orice sistem care are instalat un browser (Internet Explorer, Mozilla Firefox, Opera, etc) şi Flash Player instalat pentru acel browser. Pentru a rula aplicația în browser se foloseşte fişierul index.html.
Deasemenea, putem rula aplicația dacă este instalată o versiune Stand Alone a aplicației Flash Player. Pentru aceasta vom folosi fişierul central.swf.
Notă: Fişierul index.html depinde de central.swf, întrucât acesta include pe cel din urmă.

Detalii tehnice de implementare

O astfel de aplicație necesită o exetensibilitate mărită, deoarece oricând este posibil ca unele corecturi să fie operate sau ca noi informații să fie propuse pentru lecții. Astfel, am gândit, înainte de a crea lecțiile propriu zis, anumite module care au fost apoi folosite în toate lecțiile. Deoarece aceste module sunt dezvoltate centralizat, în momentul în care unul din ele este actualizat, schimbările vor fi reflectate în toate lecțiile. Vom prezenta în continuare unul dintre modulele dezvoltate.

Modulul UGraph

Abstractizează grafurile ca şi element al unei lecții, oferind o modalitate de a le vizuaza sau edita. Acest modul încarcă un obiect XML generat sau preluat dintr-un fişier. Acesta depinde de mai multe clase, pe niveluri de abstractizare. Iată interfețele a două dintre ele:

```
class CGraph extends MovieClip {

  var urlToGraph;
  private var data;

  function CGraph();

  var onGraphLoad:Function;
```

```
  function loadGraph(wGraph);
  private function onDataReady();

  var eventList;
  private function trigger(wEvent);

  function generate(wMinNodes, wMaxNodes, wXOffset, wYOffset, wXMLString);
}

class CGraphDisplay extends CGraph {

  var area;
  var masked;

  function CGraphDisplay();

  var drawAssetKind;

  function onDataReady();
  function onDataCreateShorts();

  function getAbsoluteX(wX);
  fuction getAbsoluteY(wY);
  function getRelativeX(wX);
  function getRelativeY(wY);

      /*+ Icons */
  function getIcon(iIconID);

      /*+ Nodes */
  function drawNodes(oneNode);
  function createNewNode(nNodeAttributes);
  function removeNode(wNode);

      /*+ Links */
  function drawLinks(oneLink);

  var pLinkAttributes;
  function createNewLink(nLinkAttributes);
  function removeLink(wLink);
  function getLink(wLinkAttributes);

      /*- generate */
  function generate(wMinNodes, wMaxNodes, wXOffset, wYOffset, wXMLString);
  function overlapping();
}
```

Funcția *generate* are două atribuții diferite în cele două clase. În clasa CGraph aceasta generează un graf XML, pe când în clasa CGraphDisplay acesta poziționează elementele vizuale şi verifică dacă acestea se suprapun. Funcția din CGraphDisplay apelează înainte funcția din CGraph, pentru a avea un suport XML. În cazul în care unele elemente vizuale ale grafului se suprapun, acesta va fi regenerat.

Structura unui fişier XML pentru grafuri

```
<graph>
  <assets>
    <icon id="1" uri="icComputer" attach="true" />
    <edge id="1" uri="egStraight" attach="true" />
  </assets>
  <nodes defaulticon="1">
      <node name="1" id="1" x="72" y="14" />
      <node name="2" id="2" x="87" y="73" />
      <node name="3" id="3" x="10" y="87" />
```

```
      <node name="4" id="4" x="34" y="25" />
      <node name="5" id="5" x="73" y="45" />
      <node name="6" id="6" x="43" y="87" />
  </nodes>
  <links defaultedge="1">
      <link nodex="5" nodey="2" cost="10" />
      <link nodex="3" nodey="6" cost="10" />
      <link nodex="6" nodey="2" cost="15" />
      <link nodex="6" nodey="5" cost="12" />
      <link nodex="4" nodey="1" cost="10" />
      <link nodex="1" nodey="5" cost="5" />
      <link nodex="4" nodey="5" cost="7" />
      <link nodex="3" nodey="4" cost="19" />
  </links>
</graph>
```

Secțiunea *assets* conține referințe către elementele vizuale ce vor fi folosite în afişarea grafului. În cazul nostru se referențiază un icon de computer şi o muchie de tip drept.

Secțiunea *nodes* conține nodurile. Fiecare nod trebuie să aibă un nume (vizibil pe ecran), un id (folosit intern pentru referențierea nodului) şi două coordonate x şi y pentru poziționarea pe ecran. Valorile pentru x şi y sunt între 0 şi 100, acestea fiind apoi scalare la dimensiunea necesară.

Secțiunea *links* conține muchiile. Fiecare muchie de la nodul x la nodul y are nevoie de specificarea nodului x şi a nodului y (prin id-ul acestora, nu prin nume)

5. Bibliografie

- **Mateescu (Cerchez) E., Maxim I.**, Arbori, Editura Țara Fagilor, Suceava, 1996
- **Horowitz E., Sahni S., Anderson-Freed S.**, Fundamentals of Data Structures in C, Computer Science Press, New York, 1993
- **Cormen Th., Leiserson Ch., Rivest R.,** Introduction to Algorithms, MIT, 1990
- **Cerchez Em.,** Informatica, Culegere de probleme pentru liceu, Editura Polirom, Iaşi, 2002