

**MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII
AL REPUBLICII MOLDOVA**

Centrul de Excelență în Informatică și Tehnologii Informaționale



RAPORT

LA PRACTICA DE INSTRUIRE

SPECIALITATEA "Administrarea aplicațiilor WEB"

TEMA : "Teren minat"

A elaborat elevul:

Iachimova Diana W-1821

Conducătorul practicii:

Frunza Olga

Chișinău 2020

Cuprins

1.Introducere	3
2.Enunțul problemei	4
3.Listingul programului	5
4.Rezultatele testării subprogramelor	14
Date de intrare:	14
Meniul:	14
Punctul 1:.....	15
Punctul 2:.....	15
Punctul 3:.....	17
Punctul 4:.....	18
Punctul 5:.....	18
Punctul 6:.....	18
Punctul 7:.....	19
Punctul 8:.....	19
Punctul 9:.....	19
5.Concluzii	20
6.Bibliografie	20

1.Introducere

O componentă de primă importanță în pregătirea viitorilor specialiști în domeniu este pregătirea practică a elevilor ce asigură conexiunea instruirii teoretice cu activitatea de producție.

În cadrul lecțiilor de programare elevii se familiarizează cu programarea în baza elaborării algoritmilor de rezolvare a unor probleme concrete simple. Astfel, este posibilă examinarea doar a principiilor generale de elaborare a programelor și anumitor aspecte ale rezolvării problemelor. Însă, rezolvarea problemelor reale presupune elaborarea unor produse computerizate mari, constituită dintr-o gamă întreagă de etape: proiectarea sistemului, elaborarea părților componente ale algoritmului, reunirea diverselor fragmente ale proiectului într-un produs final, documentarea etc. În acest context, practica de instruire preconizează imitarea întregului proces de elaborare a unui produs program, permite elevilor să evolueze în rolul de elaborator și organizator al proiectului.

Realizarea practicii de instruire vizează formarea și dezvoltarea competențelor profesionale, accentul instruirii fiind pus pe formarea de competențe digitale. Aceasta creează posibilități de dezvoltare a creativității elevilor, a gândirii critice, educând personalități social-active, capabile să rezolve problemele pe care le vor întâlni.

Obiectivele generale ale practicii de instruire:

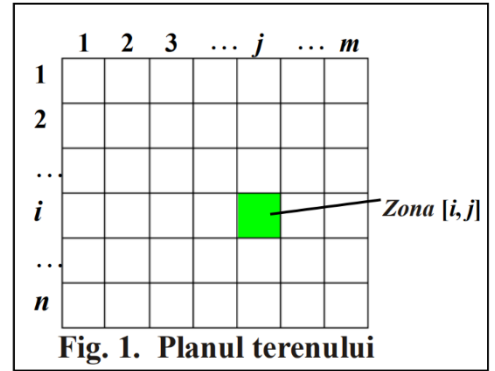
- consolidarea cunoștințelor teoretice, obținute de elevi pe parcursul studierii limbajului de programare de nivel înalt;
- aplicarea tehnicilor de programare, a elementelor de teoria grafurilor la elaborarea programelor;
- aplicarea tehnologiilor de creare și prelucrare a imaginilor;
- dezvoltarea abilităților de a lucra individual;
- formarea deprinderilor de cercetător.

Astfel practica de instruire constituie o primă lucrare complexă de sine stătătoare a elevilor folosind programarea vizuală și va contribui la asimilarea calitativă a disciplinelor ulterioare, își va aduce aportul în formarea și dezvoltarea calităților strict necesare nu numai viitorilor specialiști în domeniu, dar și fiecărui om cult care, la sigur, va trăi și va activa într-un mediu bazat pe cele mai moderne tehnologii informaționale.

2.Enunțul problemei

Planul unui teren minat, având forma unei table dreptunghiulare de dimensiunea $n \times m$ ($n, m \leq 50$) este împărțit în zone pătrate de lungimea 1 (vezi figura 1). În fiecare zonă reală a terenului poate fi plasată o mină.

Informații mai concrete despre terenul în studiu sunt înregistrate în fișierul text **Teren.in**, care conține pe prima linie numerele naturale n și m , separate prin spațiu. Fiecare din următoarele n linii ale acestui fișier conține câte m cifre binare, separate prin câte un spațiu – elementele unei matrice T , în care $T[i,j]=1$, dacă zona $[i,j]$ conține o mină, și $T[i,j]=0$ – dacă zona este liberă.

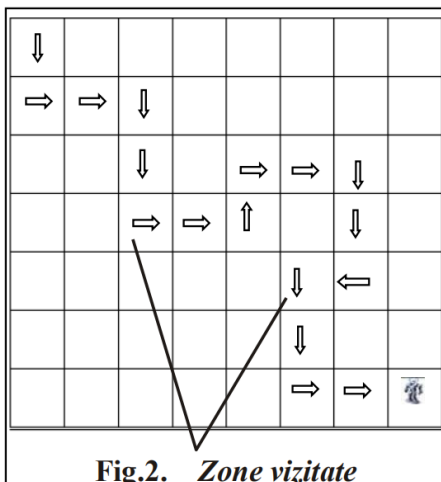


Să se elaboreze un program care, folosind meniuri și subprograme, să realizeze, la solicitarea utilizatorului, următoarele prescripții:

- 1) Înscrie în planul terenului un nou rând (marginal) / o nouă coloană (marginală); alternativa aleasă și poziția rândului (nord/sud) / coloanei (vest/est) de înscris se vor preciza de la tastatură;
- 2) „Demină” zonele unui rând / unei coloane; alternativa aleasă și numărul de ordine al rândului / coloanei de „deminat” se va preciza de la tastatură;
- 3) Determină numărul liniei/coloanei cu un număr minimal de zone minate;
- 4) Determină media numerelor de zone minate de pe coloanele pare ale terenului în studiu;
- 5) Afișează pe ecran lista numerelor de ordine ale liniilor terenului în ordinea ascendentă a numărului total de mine plasate pe liniile respective; datele se vor sorta prin metoda selecției;
- 6) Creează fișierul text **Mine.txt**, în care se vor copia doar acele linii ale fișierului de intrare **Teren.in**, care nu conțin mine;
- 7) Determină numărul de obiecte din matricea binară **T**.

Notă. Un obiect este format din elemente cu valoarea 1 care se învecinează pe linii, pe coloane sau pe diagonale.

- 8) **Rezolvă problema.** Un soldat, având la dispoziție un detector de mine, pornește dintr-un colț al terenului și trebuie să ajungă în colțul opus. Pe terenul considerat, soldatul se poate deplasa doar ortogonal și, evident, fără a nimeri în careva din zonele minate.



Să se afle traseul cel mai scurt, ce trebuie parcurs de soldat pentru a ajunge din zona $[1, 1]$, considerată, evident, neminată, în zona $[n, m]$.

Date de intrare. Informațiile despre dimensiunile terenului și zonele minate din teritoriu se conțin în fișierul text **Teren.in**, descris anterior.

Date de ieșire. Se va afișa pe ecran drumul găsit, descris prin coordonatele zonelor respective.

De exemplu, pentru ilustrarea din figura 2, drumul parcurs se va afișa astfel: [1, 1]–[2, 1]–[2, 2]– [3, 2]–[3, 3]–[4, 3]–[4, 4]–[4, 5]–[3, 5]–[3, 6]–[3, 7]– [4, 7]– [5, 7]–[5, 6]– [6, 6]–[7, 6]–[7, 7]–[7, 8]

3.Listingul programului

```
import java.util.*;
import java.io.*;
import java.text.*;

class TerenMinat{

    static DecimalFormat df=new
    DecimalFormat("0.00");

    static Scanner sc=new
    Scanner(System.in);

    static Scanner filescan=new
    Scanner(System.in);

    //citirea matricei din fisier
    static int[][] citire() {

        int n=0, m=0;

        int [][]a=null;

        try { filescan =new Scanner (new
        FileReader("Teren.in"));

        while(filescan.hasNext()) {

            //citirea dimensiunii matriciei

            n=filescan.nextInt();

            m=filescan.nextInt();

            if(n<0 || m<0) throw new
            InputMismatchException();

            a= new int[n][m];

            for(int i=0; i<n; i++)
            for(int j=0; j<m; j++) {

                //citirea elementelor matriciei

                a[i][j]=filescan.nextInt();

            }

            catch(FileNotFoundException ex)
            {System.out.println(" Fisier
            absent");}

            catch(InputMismatchException ex)
            {System.out.println(" Matricea nu
            poate contine deimensiuni
            negative");}

            return a;}

        //afisarea matricei

        static void afisare(int [][] a)

        {for(int i=0; i<a.length; i++)

        {for(int j=0; j<a[i].length; j++)

        System.out.print(a[i][j]+" ");

        System.out.println();}

        }

        //inscrierea in fisierul Teren a
        matricei modificate

        static void scriere (int a[][]) {

            int n=a.length;

            int m=a[0].length;

            try {FileWriter fw = new
            FileWriter("Teren.in");

            fw.write(n+ " "+m+"\n");

            //inscrierea dimensiunii noii
            matrici

            for(int i=0; i<a.length; i++)

            {for(int j=0; j<a[i].length; j++)

            { fw.write(a[i][j]+" ");}

            //inscrierea elementelor

            fw.write("\r\n");}

            fw.close();}

            catch (IOException ex) {

            System.out.println("Eroare de
            intrare/iesire");}

            }

            //inserarea unui rand(punctul 1)

            public static int[][]
            insertRow(int[][] m, int pozitia,
            int[] rand)

            {

                //cream o matrice noua care va stoca
                cu un rand mai mult decat matricea
                sursa
```

```

//inseram toate randurile in matricea
noua pana la indexul pozitiei noului
rand inserat

int[][] out = new int[m.length +
1][];

for (int i = 0; i < pozitia; i++) {
out[i] = m[i]; }

//inseram randul pe pozitia aleasa
//apoi sunt inerate randurile
ramase(aflata dupa indexul pozitiei
noului rand inserat)

out[pozitia] = rand;

for (int i = pozitia + 1; i <
out.length; i++) {

out[i] = m[i - 1];}

return out;}

//inserarea unei coloane (punctul 1)

public static int[][]
addColumn(int[][] a, int[] coloana,
int pozitie) {

//cream o matrice noua care va stoca
cu o coloana mai mult decat matrice
sursa

int[][] result = new
int[a.length][a[0].length + 1];

//se va copia fiecare coloana din
matricea sursa incepand cu indexul 0

//pana la indexul pozitiei (unde va
fi noua coloana)

for (int i = 0; i < a.length; i++) {

System.arraycopy(a[i], 0, result[i],
0, pozitie);

//inseram coloana pe pozitia aleasa

result[i][pozitie] = coloana[i];

//copiem coloanele ramase(aflata dupa
indexul pozitiei noii coloane
insestate)

System.arraycopy(a[i], pozitie,
result[i], pozitie + 1, a[0].length -
pozitie);}

return result;}

//citirea unui rand

```

```

static int[] citim_rand(int [][] a)
{

//numarul elementelor este egal cu
nr. coloanelor din matricea in care
va fi adaugat randul

int b[]=new int[a[0].length] ;

for(int i=0; i<a[0].length; i++)

try{
{System.out.println("b["+i+"]=");

b[i]=sc.nextInt();

if(b[i]>1 && b[i]<0) throw new
InputMismatchException(); }}

catch(InputMismatchException ex)
{System.out.println("Randul nou din
matrice poate contine doar valorile 0
sau 1"); }

return b;}

//citirea unei coloane

static int[] citim_coloana(int [][]a)
{
//numarul elementelor este egal cu
nr. liniilor din matricea in care va
fi adaugata coloana

int b[]=new int[a.length] ;

for( int i=0; i<a.length; i++)

try{
{System.out.println("b["+i+"]=");

b[i]=sc.nextInt(); if(b[i]>1 ||
b[i]<0) throw new
InputMismatchException(); }}

catch(InputMismatchException ex)
{System.out.println("Coloana noua din
matrice poate contine doar valorile 0
sau 1"); }

return b;}

//inserarea unui rand/coloane
marginal(e)

static void inserare(int [][]a) {

//definim 2 vectori care vor stoca
randul/coloana inserata

int [] r;

int []c;

String g;

```

```

System.out.println("Ce dorim sa
inseram coloana/rand");

try { g=sc.next();

if(g.contentEquals("rand")) {

System.out.println("Pozitia inserarii
nord/sud");

String k=sc.next();

if(k.contentEquals("nord")) {

r=citim_rand(a); a=insertRow(a,0,r);}

else if(k.contentEquals("sud"))
{r=citim_rand(a); a=insertRow(a,
a.length, r); }

}

else if(g.contentEquals("coloana")) {

System.out.println("Pozitia inserarii
vest/est");

String k; k=sc.next();

if(k.contentEquals("vest")) {
c=citim_coloana(a);
a=addColumn(a,c,0); }

else if(k.contentEquals("est")) {

c=citim_coloana(a); a=addColumn(a,c,
a[0].length); } } }
catch(InputMismatchException ex)
{System.out.println("Introducere
valoare gresita"); }

System.out.println("_____
_____
" );

System.out.println("Afisarea matricei
schimbate" );

afisare(a); scriere(a);

System.out.println("_____
_____
" ); }

//determinarea zonelor unui
rand/coloana

static void zone(int [][] a) {

System.out.println("Determinarea
zonelor unui rand/coloana");

try { String g=sc.next();

if(g.contentEquals("rand")) {

```

```

System.out.println("Insearati numarul
de ordine a randului de la "+1+" pana
la "+a.length );

int r=sc.nextInt();

System.out.println("Randul ales
este");

for(int i=0; i<a.length; i++)

{for(int j=0; j<a[i].length; j++)

{ if(i==r)

//afisarea randului a carui nr de
ordine coincide cu cel citit de la
tastatura
System.out.print(a[r-1][j]+" ");}}
System.out.println();

System.out.println("Zonele");

for(int j=0; j<a[0].length; j++)

{if(a[r-1][j]==1)
System.out.println("zona contine o
mina ");

else if(a[r-1][j]==0)
System.out.println("zona libera"); }

}

else if(g.contentEquals("coloana"))
{System.out.println("Insearati
numarul de ordine a coloanei "+1+"
pana la "+a[0].length);

int r; r=sc.nextInt();

System.out.println("Coloana aleasa
este");

for (int i=0; i<a.length; i++)

{for(int j=0; j<a[i].length; j++)

{ if(j==r)

//afisarea coloanei a carei nr de
ordine coincide cu cel citit de la
tastatura
System.out.print(a[i][r-1]+ " ");} }

System.out.println();

System.out.println("Zonele");

for(int i=0; i<a.length; i++)

{if(a[i][r-1]==1)
System.out.println("zona contine o
mine");

```

```

else if(a[i][r-1]==0)
System.out.println("zona libera"); }
}
}

catch(InputMismatchException ex)
{System.out.println("Introducere
valoare gresita");}

catch(ArrayIndexOutOfBoundsException
ex) {System.out.println("Nu exista
asa index");}

}

//punctul 3

//numarul randului cu un numar
minimal de zone

static void rand_min_mine(int [][] a)

{int n=a.length;

int m=a[0].length;

int [] row_arr = new int[n];

int z=0;

int row_ind=0;

for( int i=0; i<n; i++)

{ int nr=0;

for(int j=0; j<m; j++)

{if(a[i][j]==1) {nr++;} } //se
calculeaza nr. minelor de pe fiecare
linie

row_arr[z++]=nr; //se stocheaza in
vector numarul minelor de pe fiecare
linie

}

//presupunem ca prima linie are cel
mai mic nr. de mine

//se compara cu nr. de mine de pe
celelalte linii

//daca exista linie cu un nr. mai mic
de mine, se salveaza nr. de mine si
indexul liniei

int temp_row=row_arr[0];

for( int i=0; i<n; i++)

{ if(temp_row>row_arr[i])

```

```

{temp_row=row_arr[i];

row_ind=i;} }

for( int i=0; i<n; i++)

{ if(temp_row==row_arr[i])

{temp_row=row_arr[i];

row_ind=i;}

else if(temp_row!=row_arr[i])
continue;

System.out.println("Cele mai putine
mine se afla pe linia "+(row_ind+1));
}}

//punctul 3

//numarul coloanei cu un numar
minimal de zone

static void coloana_min_mine(int [][]
a)

{int n=a.length;

int m=a[0].length;

int [] col_arr = new int[m];

int y=0;

for(int j=0; j<m; j++)

{ int nr=0;

for( int i=0; i<n; i++)

{ if(a[i][j]==1) {nr++;} } //se
calculea nr de mine de pe fiecare
coloana

col_arr[y++]=nr;} //se stocheaza in
vector numarul minelor de pe fiecare
coloana

//presupunem ca prima coloana are cel
mai mic nr. de mine

//se compara cu nr. de mine de pe
celelalte coloane

//daca exista coloana cu un nr. mai
mic de mine, se salveaza nr. de mine
si indexul coloanei

int temp_col=col_arr[0];

int col_ind=0;

for( int i=0; i<m; i++)

{ if(temp_col>col_arr[i])

```



```

{temp_col=col_arr[i];
col_ind=i;}}
for( int i=0; i<m; i++)
{ if(temp_col==col_arr[i])
{temp_col=col_arr[i];
col_ind=i;}
else if (temp_col!=col_arr[i])
continue;

System.out.println("Cele mai putine
mine se afla pe coloana
" +(col_ind+1));}}

//punctul 4
//Media numerelor de zone minate pe
coloanele pare

static double average(int a[][])
{int n=a.length;
int m=a[0].length;
int [] nrpar = new int[m];
int h=0;

for(int j=1; j<m; j+=2)
{ int nr=0;

for(int i=0; i<n; i++)

{if(a[i][j]==1) {nr++;} }//se
calculeaza nr de mine de pe coloanele
pare

nrpar[h++]=nr;//se stocheaza in
vector numarul minelor de pe fiecare
coloana para
}

int suma=0, c=0;
for( int i=0; i<nrpar.length; i++)
{if(nrpar[i]!=0)

{suma=suma+nrpar[i]; //se calculeaza
numarul de zone minate

c++;}}// se calculeaza nr de coloane
pare

return (double)suma/c;}

```

```

//punctul 5

//lista numerelor de ordine ale
liniilor terenului in ordinea
ascendenta a numerelor de mine

static void sortare(int [][] a) {

int n=a.length;

int m=a[0].length;

int [] numar = new int[n];

int r=0;

int [] index= new int[n];

int o=0;

for( int i=0; i<n; i++)

{ int nr=0;

for(int j=0; j<m; j++)

{if(a[i][j]==1) {nr++;}}//se
calculeaza nr de mine pe fiecare line

numar[r++]=nr; //se stocheaza in
vector numarul minelor de pe fiecare
linie

index[o++]=i+1;}//se stocheaza in
vector indexul fiecărei linii

int aux=0; int aix=0;

//sortarea

for( int i=0; i<numar.length-1; i++)

{ int iMin=i;//iMin- idexul minim

for(int v=i+1; v<numar.length; v++)

if(numar[v]<numar[iMin])//daca al
doilea nr. e mai mic ca primul,
acestea se vor interchimba

iMin=v;

aux=numar[iMin];
numar[iMin]=numar[i]; numar[i]=aux;

//interschimbarea indexului

aix=index[iMin];
index[iMin]=index[i]; index[i]=aix;}

for( int i=0; i<a.length; i++)

System.out.println(index[i]);}

```

```

//subpunctul 6

```

```

//copierea in fisier liniile care nu
au mine

static void scriere_mine (int [][] a)
{
    try { FileWriter fw = new
        FileWriter("Mine.txt");

        for(int i=0; i<a.length; i++)

            {int nr=0;

            for(int j=0; j<a[i].length; j++)

                if(a[i][j]==0) {nr++;} //nr de mine de
                pe fiecare line

                int m=a[i].length;

                if(nr==m) //daca intreaga linie este
                fara mine se va inscrie in fisier

                for(int j=0; j<a[i].length; j++)

                    {fw.write(a[i][j]+" ");}

                    fw.write("\r\n");} fw.close();}
                catch (IOException ex) {
                System.out.println("Eroare de
                intrare/iesire");}

            }

        //determinarea daca exista macar un
        vecin (punctul 7)

        static int calcul(int[][] a, int i,
        int j, int current_count, int n, int
        m) {

            a[i][j] = 0; //elementul se va nota
            cu 0 pentru a nu fi vizitat de doua
            ori

            //daca exista o coordonata valida, se
            verifica daca aceasta are vecini

            //daca exista cel putin un
            vecin(obiect) se va returna 1

            if (verificare(i-1,j , a.length,
            a[0].length) == 1 && a[i-1][j] == 1)
            {calcul(a,i-1,j ,current_count,n,m);
            current_count = 1;}

            if (verificare(i+1,j , a.length,
            a[0].length) == 1 && a[i+1][j] == 1)
            {calcul(a,i+1,j ,current_count,n,m);
            current_count = 1;}

            if (verificare(i,j-1, a.length,
            a[0].length) == 1 && a[i][j-1] == 1)
            {calcul(a,i, j-1,current_count,n,m);
            current_count = 1;}

```

```

            if (verificare(i,j+1, a.length
            ,a[0].length) == 1 && a[i][j+1] == 1)
            {calcul(a,i ,j+1,current_count,n,m);
            current_count = 1;}

            if (verificare(i+1,j-1, a.length,
            a[0].length) == 1 && a[i+1 ][j-1] ==
            1) {calcul(a,i+1 ,j-
            1,current_count,n,m); current_count =
            1;}

            if (verificare(i+1,j+1, a.length,
            a[0].length) == 1 && a[i+1][j+1] ==
            1) {calcul(a,i+1
            ,j+1,current_count,n,m);
            current_count = 1;}

            if (verificare(i-1,j-1, a.length,
            a[0].length) == 1 && a[i-1][j-1] ==
            1) {calcul(a,i-1 ,j-
            1,current_count,n,m); current_count =
            1;}

            if (verificare(i-1,j+1, a.length,
            a[0].length) == 1 && a[i-1][j+1] ==
            1) {calcul(a,i-1
            ,j+1,current_count,n,m);
            current_count = 1;}

            return current_count;}

        //verificarea daca coordonatele nu
        depasesc limitele matricei (punctul
        7)

        static int verificare(int i, int j,
        int n, int m) {

            if (i >= 0 && i <n && j >= 0 && j <
            m) {return 1;}

            else {return 0;} }

        //subpunctul 7

        //determinarea numarului de obiecte

        static int nr_obiecte(int [][] a) {

            int n=a.length;

            int m=a[0].length;

            int count=0;

            int current_count;

            for(int i=0; i<n; i++) {

                for(int j=0; j<m; j++) {

                    if (a[i][j] == 1) {

                        current_count = 0;

```

```

count = count + calcul(a,i,j,
current_count, n,m); }

}} return count; }

//meniul

static void menu() {
Scanner sc=new Scanner(System.in);
int [][] a;
a=citire();

System.out.println("-----
-----");

System.out.println("          -----
-----");

System.out.println("
-----");

System.out.println("
-----");

System.out.println("
-----");

System.out.println("          -----
-----");

System.out.println("-----
-----");

System.out.println(" 1.Citirea
matricei din fisier si afisarea
acestea ");

System.out.println(" 2.Inscrierea in
planul terenului un nou rand/colana
marginala");

System.out.println(" 3.Determinarea
zonelor unui rand/unei coloane");

System.out.println(" 4.Determinarea
numarului liniei/cloanei cu un nr.
minimal de zone");

System.out.println(" 5.Determinarea
mediei numerelor de zone minate de pe
coloanele pare");

System.out.println(" 6.Afisarea
listei numerelor de ordine ale
liniilor terenului in ");

```

```

System.out.println("  ordine
ascendenta a numarului total de
mine");

System.out.println(" 7.Copierea
liniilor care nu contin mine in alt
fisier");

System.out.println(" 8.Determinarea
numarului de obiecte din matrice");

System.out.println(" 9.Problama");

System.out.println(" 0.Exit");

System.out.println("-----
-----");

System.out.println("Alegerea dvs.");

int v=0;

try {v=sc.nextInt();}

catch(InputMismatchException ex) {
System.out.println("Valoare
gresita"); menu();}

switch(v) {

case 0: break;

case
1:System.out.println("_____
_____");

System.out.println("Matricea
este:");afisare(a) ;

System.out.println("_____
_____"); break;

case
2:System.out.println("_____
_____");

inserare(a);break;

case 3:
System.out.println("_____
_____");

zone(a);
System.out.println("_____
_____"); break;

case
4:System.out.println("_____
_____");

```

```

coloana_min_mine(a);
System.out.println("_____
_____
" );
rand_min_mine(a);
System.out.println("_____
_____
" );break;

case
5: System.out.println("_____
_____
" );

double l; l=average(a);
System.out.println("Media este "+
df.format(l));

System.out.println("_____
_____
" ); break;

case 6:
System.out.println("_____
_____
" );

System.out.println("Lista numerelor
de ordine a liniilor sortate:");
sortare(a);

System.out.println("_____
_____
" ); break;

case
7: System.out.println("_____
_____
" );

scriere_mine(a);
System.out.println("Datele au fost
inscrise");

System.out.println("_____
_____
" ); break;

case
8: System.out.println("_____
_____
" );

int count=nr_obiecte(a);
System.out.println("Numarul de
obiecte din matrice e "+ count);

System.out.println("_____
_____
" );break;

case 9: CelMaiScurDrum.print(a);

```

```

System.out.println("_____
_____
" );break;

default: System.out.println("Ati
introdus valoarea gresita");}

System.out.println("Continuare
true/false");

boolean f = true;

try {f=sc.nextBoolean();}

catch(InputMismatchException ex) {
System.out.println("Valoare
gresita");}

catch(NoSuchElementException ex) {
System.out.println("Nu exista asa
elemnt");}

if(f) menu();

sc.close();}

public static void main(String[]
args) { menu();}}

class CelMaiScurDrum {
static class Cell {

int i;

int j;

int dist; //va memora distanta de la
pozitia curenta la sursa

Cell prev; //va memora ultimul
element vizitat

//constructor parametrizat

Cell(int i, int j, int dist, Cell
prev) {

this.i = i;

this.j = j;

this.dist = dist;

this.prev = prev;}

//returneaza un string a obiectului
specificat, in cazul nostru
coordonatele elementelor din coada

public String toString(){

return "("+(i+1)+ ", "+(j+1)+")";} }

```

```

//afiseaza cel mai scurt drum

public static void print(int[][] a) {

    if (a[0][0] == 1 || a[a.length-1][a[0].length-1] == 1) return;
    //daca nu exista sursa sau destinatie
    //iese din program

    Cell[][] cells = new
    Cell[a.length][a[0].length];

    for (int i = 0; i < cells.length;
    i++) {

        for (int j = 0; j < cells[0].length;
        j++) {

            if (a[i][j] != 1) {

                cells[i][j] = new Cell(i, j,
                Integer.MAX_VALUE, null);
                //initializarea }}

            }

            LinkedList<Cell> queue = new
            LinkedList<>();

            Cell src = cells[0][0]; //sursa

            src.dist = 0;

            queue.add(src); //adaugam sursa in
            coada

            Cell dest = null; //va stoca
            destinatia daca va fi gasita

            Cell curent; //pointer catre
            elementul curent

            while ((curent = queue.poll()) !=
            null) {

                if (curent.i==a.length-1 && curent.j
                == a[0].length-1) {

                    dest = curent; //daca am ajuns la
                    destinatie orpim cautarea}

                    visit(cells, queue, curent.i - 1,
                    curent.j, curent);

                    visit(cells, queue, curent.i + 1,
                    curent.j, curent);

                    visit(cells, queue, curent.i,
                    curent.j - 1, curent);

```

```

visit(cells, queue, curent.i,
curent.j + 1, curent);}

if (dest == null) {return;

} else {

    LinkedList<Cell> path = new
    LinkedList<>();

    curent = dest; //recitim drumul
    anterior incepand cu destinatia

    do {

        path.addFirst(curent); //salvam
        coordonatele drumului

    } while ((curent = curent.prev) !=
    null);

    System.out.println(path); //afisarea

    }}

    static void visit(Cell[][] cells,
    LinkedList<Cell> queue, int i, int j,
    Cell parent) {

        int dist = parent.dist + 1;
        //incrementam distanta

        if (i < 0 || i >= cells.length || j <
        0 || j >= cells[0].length ||
        cells[i][j] == null) {

            return; //daca coordonatele depasesc
            limitele matricei, iesire din functie

        }

        Cell curent = cells[i][j];

        if (dist < curent.dist) {

            curent.dist = dist;

            curent.prev = parent;

            queue.add(curent); //adaugam
            elementul valid in coada

        }

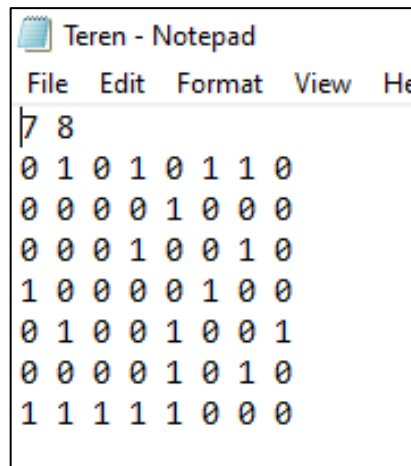
    }

}

```

4.Rezultatele testării subprogramelor

Date de intrare:

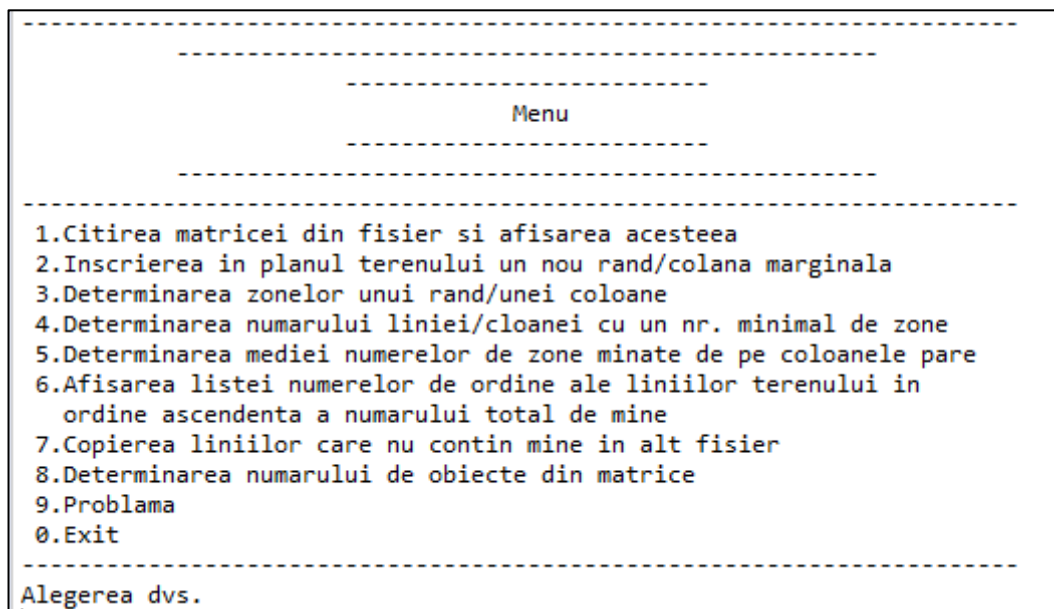


```
Teren - Notepad
File Edit Format View Help
7 8
0 1 0 1 0 1 1 0
0 0 0 0 1 0 0 0
0 0 0 1 0 0 1 0
1 0 0 0 0 1 0 0
0 1 0 0 1 0 0 1
0 0 0 0 1 0 1 0
1 1 1 1 1 0 0 0
```

Imag.1: Fișier Teren.in

Fișierul Teren.in stochează informația despre terenul minat în studiu. Pe prima linie sunt plasate 2 numere naturale n și m , separate prin spațiu, reprezentând dimensiunea matricei (n - nr.de linii, m - nr de coloane).Fiecare din următoarele n linii ale acestui fișier conțin câte m cifre binare, separate prin câte un spațiu, reprezentând elementele unei matrice. În cazul în care elementul matricei este 1 – acesta reprezintă o zonă minată(conține o mină), dacă elementul e 0 – acesta reprezintă o zonă liberă.

Meniul:



```
-----
                        -----
                        Menu
                        -----
-----
1.Citirea matricei din fisier si afisarea acesteea
2.Inscrierea in planul terenului un nou rand/colana marginala
3.Determinarea zonelor unui rand/unei coloane
4.Determinarea numarului liniei/cloanei cu un nr. minimal de zone
5.Determinarea mediei numerelor de zone minate de pe coloanele pare
6.Afisarea listei numerelor de ordine ale liniilor terenului in
  ordine ascendenta a numarului total de mine
7.Copierea liniilor care nu contin mine in alt fisier
8.Determinarea numarului de obiecte din matrice
9.Problama
0.Exit
-----
Alegerea dvs.
```

Imag.2: Meniul

Meniul reprezintă principalul mod de interacțiune a programului cu utilizatorul. La inserarea unui număr de la 0 până la 9 programul va afișa răspuns la subpunctul corespunzător.

Punctul 1:

```
Alegerea dvs.
1

Matricea este:
0 1 0 1 0 1 1 0
0 0 0 0 1 0 0 0
0 0 0 1 0 0 1 0
1 0 0 0 0 1 0 0
0 1 0 0 1 0 0 1
0 0 0 0 1 0 1 0
1 1 1 1 1 0 0 0
```

Imag.3: Afişare

La alegerea punctului 1 se va afişa matricea citică din fişierul Teren.in

Punctul 2:

```
Alegerea dvs.
2

Ce dorim sa inseram coloana/rand
rand
Pozitia inserarii nord/sud
nord
b[0]=
0
b[1]=
0
b[2]=
0
b[3]=
0
b[4]=
0
b[5]=
0
b[6]=
0
b[7]=
0

Afisarea matricei schimbate
0 0 0 0 0 0 0 0
0 1 0 1 0 1 1 0
0 0 0 0 1 0 0 0
0 0 0 1 0 0 1 0
1 0 0 0 0 1 0 0
0 1 0 0 1 0 0 1
0 0 0 0 1 0 1 0
1 1 1 1 1 0 0 0
```

Imag.4.1:Inserarea unui rand la nord

```
Alegerea dvs.
2

Ce dorim sa inseram coloana/rand
rand
Pozitia inserarii nord/sud
sud
b[0]=
1
b[1]=
0
b[2]=
1
b[3]=
1
b[4]=
1
b[5]=
0
b[6]=
1
b[7]=
0

Afisarea matricei schimbate
0 0 0 0 0 0 0 0
0 1 0 1 0 1 1 0
0 0 0 0 1 0 0 0
0 0 0 1 0 0 1 0
1 0 0 0 0 1 0 0
0 1 0 0 1 0 0 1
0 0 0 0 1 0 1 0
1 1 1 1 1 0 0 0
1 0 1 1 1 0 1 0
```

Imag.4.2:Inserarea unui rand la sud

Alegerea dvs.
2

Ce dorim sa inseram coloana/rand
coloana

Pozitia inserarii vest/est
vest

b[0]=
0

b[1]=
1

b[2]=
0

b[3]=
0

b[4]=
1

b[5]=
1

b[6]=
1

b[7]=
0

b[8]=
0

Afisarea matricei schimbate

File	Edit	Format	View
9 9			
0 0 0 0 0 0 0 0 0			
0 1 0 1 0 1 1 0 0			
0 0 0 0 1 0 0 0 0			
0 0 0 1 0 0 1 0 0			
1 0 0 0 0 1 0 0 0			
0 1 0 0 1 0 0 1 0			
0 0 0 0 1 0 1 0 0			
1 1 1 1 1 0 0 0 0			
1 0 1 1 1 0 1 0 0			

Imag.4.3:Inserarea unei coloane la vest

Alegerea dvs.
2

Ce dorim sa inseram coloana/rand
coloana

Pozitia inserarii vest/est
est

b[0]=
0

b[1]=
0

b[2]=
0

b[3]=
0

b[4]=
0

b[5]=
0

b[6]=
0

b[7]=
0

b[8]=
0

Afisarea matricei schimbate

File	Edit	Format	View
9 10			
0 0 0 0 0 0 0 0 0 0			
1 0 1 0 1 0 1 1 0 0			
0 0 0 0 0 1 0 0 0 0			
0 0 0 0 1 0 0 1 0 0			
1 1 0 0 0 0 1 0 0 0			
1 0 1 0 0 1 0 0 1 0			
1 0 0 0 0 1 0 1 0 0			
0 1 1 1 1 1 0 0 0 0			
0 1 0 1 1 1 0 1 0 0			

Imag.4.4:Inserarea unei coloane la est

La alegerea punctului 2 se va realiza inserarea unui rând marginal sau coloane marginale. Utilizatorul alege ce vrea să insereze un rând sau coloană. Dacă alegerea lui e rând, îi apar două opțiuni, reprezentând poziția de inserare (nord –rândul nou va fi inserat ca prima linie(nr. de ordine 1), sud-rândul nou va fi inserat ca ultima linie). Dacă alegerea utilizatorului este coloana, îi apar două opțiuni, reprezentând poziția de inserare (vest- coloana nouă va fi inserată ca prima coloană (nr. de ordine 1); est-coloana va fi inserată ca ultima coloană din matrice). În cazul oricărei opțiuni alese după citirea rândului marginal sau coloanei marginale se afișează matricea nouă cu rândul/coloana marginal(ă) inserată, această matrice nouă este înscrisă în fișierul Teren.in, dimensiunile aflate pe prima linie a fișierului fiind actualizate, pentru lucrul în prealabil cu aceasta.

Teren - Notepad

9	10									
0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	1	0	0	
0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	1	0	0	
1	1	0	0	0	0	1	0	0	0	
1	0	1	0	0	1	0	0	1	0	
1	0	0	0	0	1	0	1	0	0	
0	1	1	1	1	1	0	0	0	0	
0	1	0	1	1	1	0	1	0	0	

Imag.4.5. Matricea de lucru

După inserarea unor noi linii și coloane marginale s-a format o matrice cu noi dimensiuni și elemente. Aceasta s-a salvat în fișierul Teren.in, iar următoarele operații vor fi realizate prin intermediul ei.

Punctul 3:

```
Alegerea dvs.
3
Determinarea zonelor unui rand/coloana
rand
Insearati numarul de ordine a randului de la 1 pana la 9
2
Randul ales este
1 0 1 0 1 0 1 1 0 0
Zonele:
zona contine o mina
zona libera
zona contine o mina
zona libera
zona contine o mina
zona libera
zona contine o mina
zona contine o mina
zona libera
zona libera
```

Imag.5.1:Determinarea zonelor unui rand

```
Alegerea dvs.
3
Determinarea zonelor unui rand/coloana
coloana
Insearati numarul de ordine a coloanei 1 pana la 10
3
Coloana aleasa este
0 1 0 0 0 1 0 1 0
Zonele:
zona libera
zona contine o mine
zona libera
zona libera
zona libera
zona libera
zona contine o mine
zona libera
zona contine o mine
zona libera
```

Imag.5.2:Determinarea zonelor unei coloane

La alegerea punctului 3 se determină zonele unui rând sau coloane. Utilizatorul alege dintre două alternative. După optarea unei alternative se va cere precizarea numărului de ordine a rândului sau coloanei afișându-se limitele indecșilor. La inserarea numărului se afișează rândul/coloana aleasă urmată de lista zonelor. Dacă elementul rândului/coloanei e 1 – se afișează “zona conține o mină “, dacă elementul rândului/coloanei e 0 – se afișează “zonă liberă“.

Punctul 4:

```
Alegerea dvs.  
4  
-----  
Cele mai putine mine se afla pe coloana 10  
-----  
Cele mai putine mine se afla pe linia 1
```

Imag.6: Nr. minimal de mine

La alegerea punctului 4 se va afișa numărul de ordine a liniei și coloanei care are cel mai mic număr de mine.

Punctul 5:

```
Alegerea dvs.  
5  
-----  
Media este 3,50  
-----
```

Imag.7:Media

La alegerea punctului 5 se afișează media numerelor de zone minate de pe coloanele pare ale terenului în studiu.

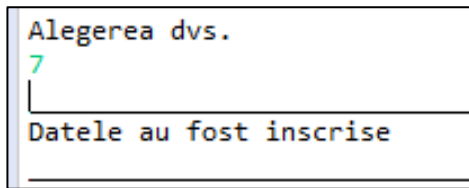
Punctul 6:

```
Alegerea dvs.  
6  
-----  
Lista numerelor de ordine a liniilor sortate:  
1  
3  
4  
5  
7  
6  
2  
8  
9
```

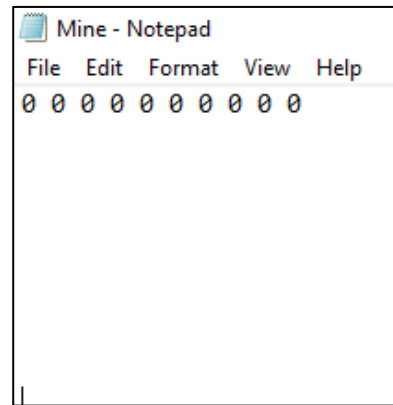
Imag.8:Sortare

La alegerea punctului 6 se afișează lista numerelor de ordine ale liniilor terenului în ordinea ascendentă a numărului total de mine plasate pe liniile respective. Astfel primul număr din lista este numărul de ordine a liniei care are cel mai mic număr de mine, iar ultimul – cel mai mare număr de mine.

Punctul 7:



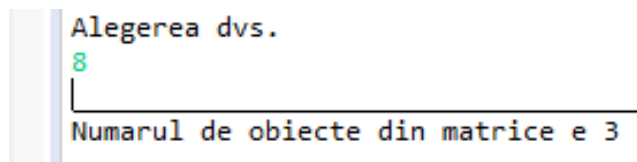
Imag.9.1: Linii fără mine



Imag.9.2: Mine.txt

La alegerea punctului 7 se creează fișierul Mine.txt în care se copiază doar acele linii ale fișierului de intrare Teren.in, care nu conțin mine. Liniile copiate își păstrează aceeași poziție asemeni ca în fișierul Teren.in. La ecran se afișează mesajul despre succesul înscrierii datelor în fișierul Mine.txt.

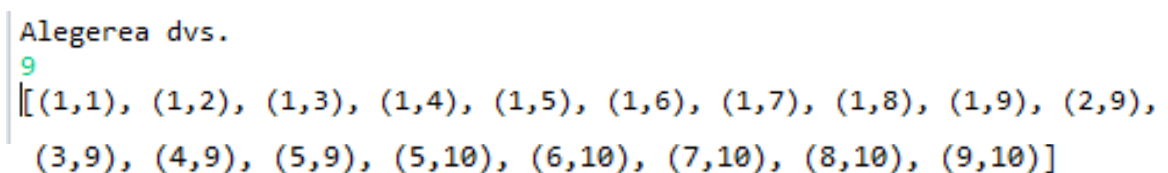
Punctul 8:



Imag.10: Obiecte

La alegerea punctului 8 la ecran se va afișa numărul de obiecte din matrice. Un obiect este format din elementele cu valoarea 1 care se învecinează pe linii, coloane sau pe diagonale. În cazul în care un element cu valoare 1 nu se învecinează acesta nu formează de sine stătător un obiect.

Punctul 9:



Imag.11: Cel mai scurt drum

La alegerea punctului 9 la ecran se va afișa traseul celui mai scurt drum, descris prin coordonatele zonelor respective, din zona [1,1] în zona [n,m], evitând zonele minate.

5.Concluzii

Pentru mine, practica de instruire reprezintă o etapă de pregătire în devenirea unui tânăr specialist. Aceasta reprezintă un program complex în care am folosit pe larg diverse tipuri de date: tablouri, fișiere, etc, antrenându-mi capacitățile de lucru cu acestea. În cadrul etapei date am aplicat principiile programării structurate, programării procedurale și programării orientate pe obiecte în rezolvarea programului. Acesta fiind format în mare parte din funcții recursive direct și indirect. Pentru realizarea punctului 8 am avut nevoie de mai mult timp, îmbinând recursivitatea, elemente din teoria grafurilor, căutarea în lățime (bfs) și stocarea informației în coadă(queue) - o structură de date logică și omogenă. Practica de instruire a jucat un rol important pentru dezvoltarea în performanță a competențelor digitale și gândirii analitice. Pentru realizarea unui produs finit a fost nevoie de răbdare, organizare și documentare, fiind constituit dintr-o gamă largă de etape. Aceasta constituie prima lucrare complexă de sine stătătoare, care mi-a permis să evoluez în rolul de elaborator și organizator al proiectului. Dificulățile întâlnite au fost depășite prin consultarea literaturii de specialitate. Posibilele cercetări viitoare pot fi similare cu această experiență. Rezultatul cercetării satisface obiectivele propuse la început.

În cadrul realizării practicii de instruire am obținut cunoștințe, mi-am format competențe specifice și profesionale, asumându-mi responsabilități, manifestând gândire critică și creativă. Aceasta reprezintă o experiență importantă și unică, care m-a format ca specialist în domeniu, mi-a oferit cunoștințe, performanța abilităților de cercetare, analizare și rezolvare.

6.Bibliografie

Literatură:

1. <http://www.oracle.com/technetwork/java/index.html>;
2. C. Olaru, S. Tanasa, Java de la 0 la expert, Polirom, Iasi, 2003;
3. B. Eckel, Thinking in Java, Prentice Hall (4-th Edition), 2006.

Site-uri:

1. <https://www.guru99.com/breadth-first-search-bfs-graph-example.html>;
2. <https://www.baeldung.com/java-round-decimal-number>;
3. <http://homepage.cs.uiowa.edu/~sriram/21/fall04/code/Matrix.java>;
4. <http://math.hws.edu/javanotes/c7/s5.html>;
5. <https://coderanch.com/t/467706/java/Sorting-arrays-retaining-original-index>;
6. <https://www.programiz.com/java-programming/algorithms>.