Ministerul Educaţiei, Culturii și Cercetării

al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Ingineria Software şi Automatică

RAPORT

Lucrarea de laborator Nr.3

la Matematica specială

#### Tema : Algoritmul lui Bellman-Calaba și Ford pentru determinarea drumului minim (maxim) .

A efectuat : gr. SI – 201 , Ivanova Evghenia

A verificat : asis. univ. Popovici Nadejda

Chişinău – 2021

**Scopul Lucrării :**

##### Studierea algoritmilor de determinare a drumurilor minime  într-un graf.

##### Studierea algoritmilor de determinare a drumurilor maxime  într-un graf.

##### Elaborarea programelor de determinare a drumului minim și maxim într-un graf ponderat cu ajutorul algoritmului Ford și algoritmului Bellman-Calaba.

**Sarcina de bază :**

1. Elaboraţi un program cu următoarele posibilităţi :

* introducerea grafului în calculator,
* determinarea drumului minim și maxim în graful ponderat cu ajutorul algoritmului Ford și algoritmului Bellman-Calaba;
* vizualizarea rezultatelor la display şi imprimantă.

**Codul programului :**

#include <stdio.h>

int main()

{

int i, j, k=1, v, min, max;

printf("\tNr de varfuri "); scanf("%d",&v);

int H[v], x[v][v], V[5][v], l[v];

for(i=1; i<=v; i++)

for(j=1; j<=v; j++)

{

if(i==j){ x[i][j]=0; printf("x%d -> x%d= %d\n",i,j,x[i][j]); }

else{ printf("x%d -> x%d= ",i,j); scanf("%d",&x[i][j]); }

}

printf("\n\t1.Drumul minim Ford\n\n");

H[1]=0; for(i=2;i<=v;i++) H[i]=100;

for(i=1; i<=v; i++) printf(" x%d ",i); printf("\n");

for(j=1; j<v; j++)

{

for(i=1;i<=v;i++)

{

if((x[j][i]!=0) && (x[j][i]!=100) && (H[i]-H[j]>x[j][i]))

{

H[i]=H[j]+x[j][i];

printf(" %d ",H[i]);

}

else printf(" %d ",H[i]);

}

printf("\n");

}

printf("\nl min = %d\t",H[v]); l[v]=v;

if(H[v]-H[v-1]==x[v-1][v]) l[v-1]=v-1;

else if(H[v]-H[v-2]==x[v-2][v]) l[v-1]=v-2;

else if(H[v]-H[v-3]==x[v-3][v]) l[v-1]=v-3;

if(H[v-1]-H[v-2]==x[v-2][v-1]) l[v-2]=v-2;

else if(H[v-1]-H[v-3]==x[v-3][v-1]) l[v-2]=v-3;

else if(H[v-1]-H[v-4]==x[v-4][v-1]) l[v-2]=v-4;

if(H[v-2]-H[v-3]==x[v-3][v-2]) l[v-3]=v-3;

else if(H[v-2]-H[v-4]==x[v-4][v-2]) l[v-3]=v-4;

else if(H[v-2]-H[v-5]==x[v-5][v-2]) l[v-3]=v-5;

if(H[v-3]-H[v-4]==x[v-4][v-3]) l[v-4]=v-4;

else if(H[v-3]-H[v-5]==x[v-5][v-3]) l[v-4]=v-5;

else if(H[v-3]-H[v-6]==x[v-6][v-3]) l[v-4]=v-6;

if(H[v-4]-H[v-5]==x[v-5][v-4]) l[v-5]=v-5;

else if(H[v-4]-H[v-6]==x[v-6][v-4]) l[v-5]=v-6;

if(H[v-5]-H[v-6]==x[v-6][v-5]) l[v-6]=v-6; l[v-7]=1;

for(i=1;i<=v;i++) if((l[i]!=0) && (l[i]!=l[i+1]) && (l[i])<=v) printf(" %d ",l[i]);

printf("\n\n\t2.Drumul maxim Ford\n\n");

H[1]=0; for(i=2;i<=v;i++) H[i]=-100;

for(j=1; j<v; j++)

{

for(i=1;i<=v;i++)

{

if((x[j][i]!=0) && (x[j][i]!=100) && (H[i]-H[j]<x[j][i]))

{

H[i]=H[j]+x[j][i];

printf(" %d ",H[i]);

}

else printf(" %d ",H[i]);

}

printf("\n");

}

printf("\n\nl max = %d\t",H[v]); l[v]=v;

if(H[v]-H[v-1]==x[v-1][v]) l[v-1]=v-1;

else if(H[v]-H[v-2]==x[v-2][v]) l[v-1]=v-2;

else if(H[v]-H[v-3]==x[v-3][v]) l[v-1]=v-3;

if(H[v-1]-H[v-2]==x[v-2][v-1]) l[v-2]=v-2;

else if(H[v-1]-H[v-3]==x[v-3][v-1]) l[v-2]=v-3;

else if(H[v-1]-H[v-4]==x[v-4][v-1]) l[v-2]=v-4;

if(H[v-2]-H[v-3]==x[v-3][v-2]) l[v-3]=v-3;

else if(H[v-2]-H[v-4]==x[v-4][v-2]) l[v-3]=v-4;

else if(H[v-2]-H[v-5]==x[v-5][v-2]) l[v-3]=v-5;

if(H[v-3]-H[v-4]==x[v-4][v-3]) l[v-4]=v-4;

else if(H[v-3]-H[v-5]==x[v-5][v-3]) l[v-4]=v-5;

else if(H[v-3]-H[v-6]==x[v-6][v-3]) l[v-4]=v-6;

if(H[v-4]-H[v-5]==x[v-5][v-4]) l[v-5]=v-5;

else if(H[v-4]-H[v-6]==x[v-6][v-4]) l[v-5]=v-6;

if(H[v-5]-H[v-6]==x[v-6][v-5]) l[v-6]=v-6; l[v-7]=1;

for(i=1;i<=v;i++) if((l[i]!=0) && (l[i]!=l[i+1]) && (l[i])<=v) printf(" %d ",l[i]);

printf("\n\n\t3.Drumul minim BK\n\n");

for(i=1; i<=v; i++) printf("\tx%d",i); printf("\n");

for(i=1; i<=v; i++)

{

printf("x%d | ",i);

for(j=1; j<=v; j++)

{

x[7][4]=100; x[7][5]=100; x[7][6]=100;

if(i==j){ x[i][j]=0; printf("%d\t",x[i][j]); }

else printf("%d\t",x[i][j]);

}

printf("\n");

}

printf("V | ");

for(i=1; i<=v; i++)

{

V[1][i]=x[i][v];

if(i==v) V[1][i]=0;

else printf("%d\t",V[1][i]);

}

for(k=2; k<=5; k++)

{

for(i=1; i<=v; i++) V[k][i]=100;

for(i=1; i<=v; i++)

for(j=1; j<=v; j++)

{

min=V[k-1][i]+x[j][i];

if(min<V[k][j]) V[k][j]=min;

}

printf("\nV%d | ",k-1);

for(i=1; i<=v; i++) printf("%d\t",V[k][i]);

}

printf("\n\nl min = %d",V[5][1]); l[1]=1; printf("\t%d ",l[1]);

for(i=2; i<=v; i++)

{

if(V[5][i-1]-V[5][i]==x[i-1][i]){ l[i]=i; printf(" %d ",l[i]);}

else if(V[5][i-1]-V[5][i+1]==x[i-1][i+1]){ l[i]=i+1; printf(" %d ",l[i]); i++;}

}

printf("\n\n\t4.Drumul maxim BK\n\n");

for(i=1; i<=v; i++) printf(" x%d",i); printf("\n");

for(i=1; i<=v; i++)

{

printf("x%d | ",i);

for(j=1; j<=v; j++)

{

if(x[i][j]==100) printf("%f ", 1/0.0);

else printf("%d ",x[i][j]);

}

printf("\n");

}

printf("V | ");

for(i=1; i<=v; i++)

{

V[1][i]=x[i][v];

if(i==v) V[1][i]=0;

if(V[1][i]==100) printf("%f ", 1/0.0);

else printf("%d ",V[1][i]);

}

for(k=2; k<=5; k++)

{

for(i=1; i<=v; i++) V[k][i]=0;

for(i=1; i<=v; i++)

for(j=1; j<=v; j++)

{

max=V[k-1][i]+x[j][i];

if((max>V[k][j]) && (max<100)) V[k][j]=max;

}

printf("\nV%d | ",k-1);

for(i=1; i<=v; i++) printf("%d ",V[k][i]);

}

printf("\n\nl max = %d",V[5][1]); l[1]=1; printf("\t%d ",l[1]);

for(i=2; i<=v; i++)

{

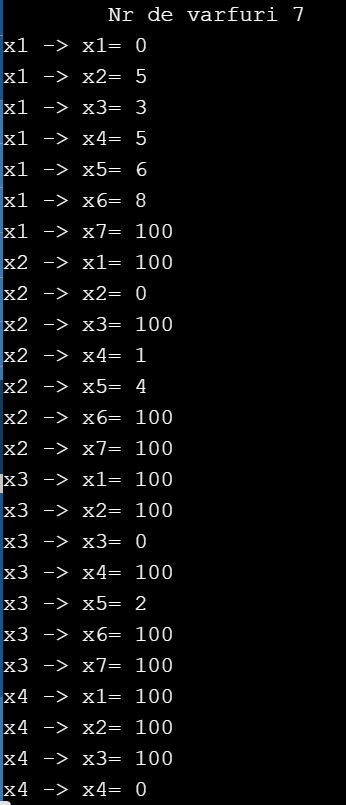
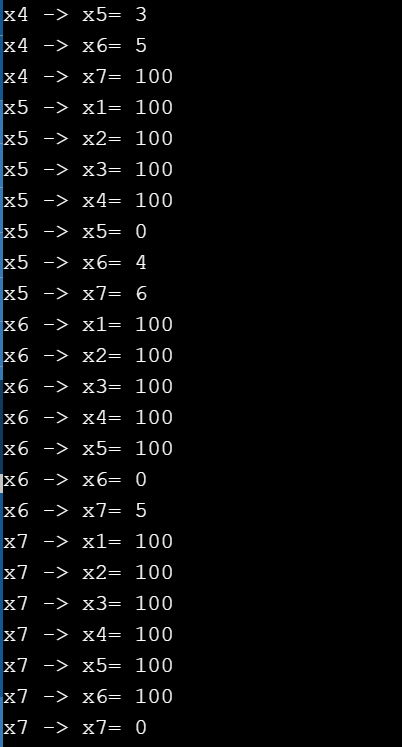
if(V[5][i-1]-V[5][i]==x[i-1][i]){ l[i]=i; printf(" %d ",l[i]);}

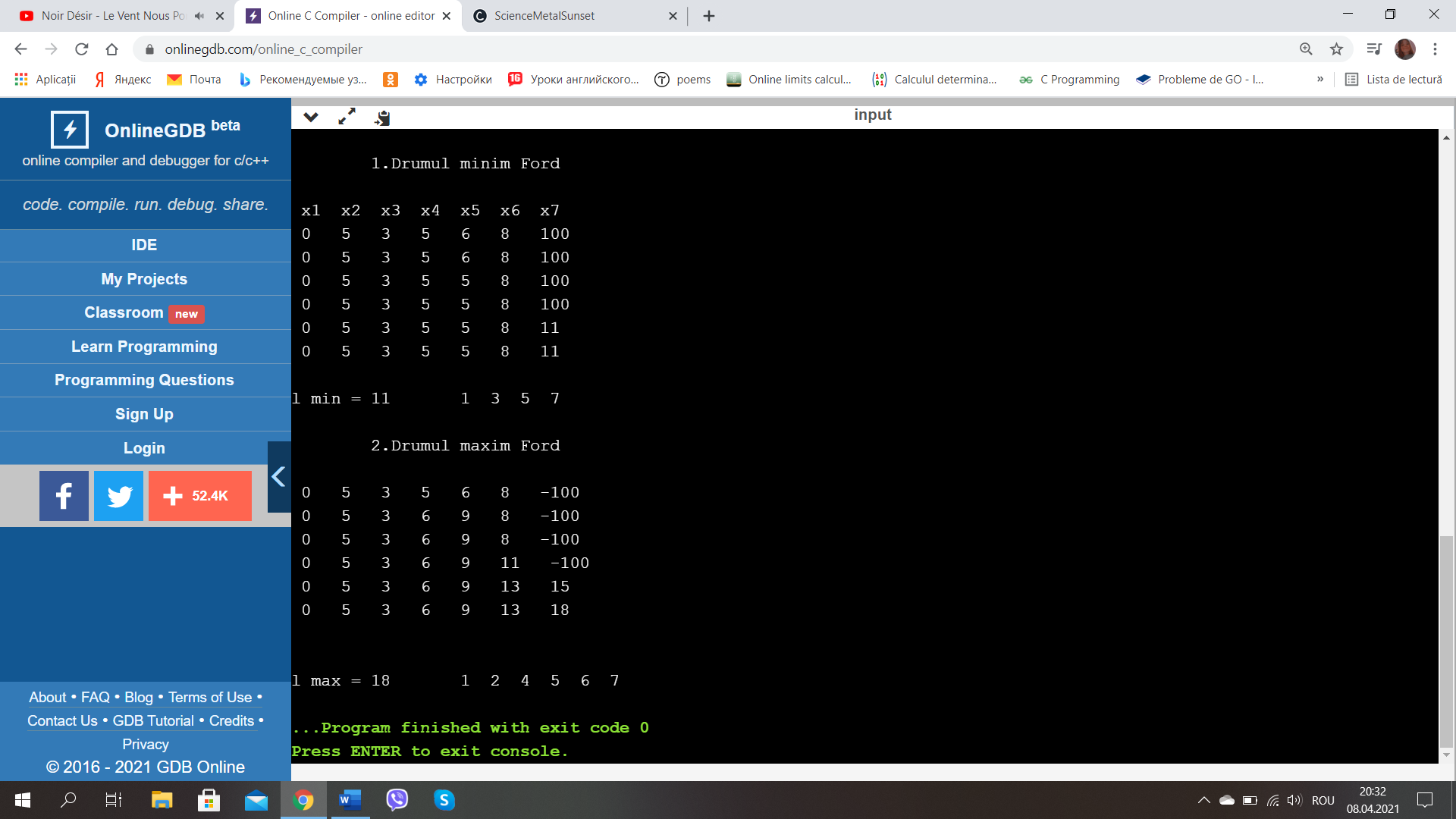
else if(V[5][i-1]-V[5][i+1]==x[i-1][i+1]){ l[i]=i+1; printf(" %d ",l[i]); i++;}

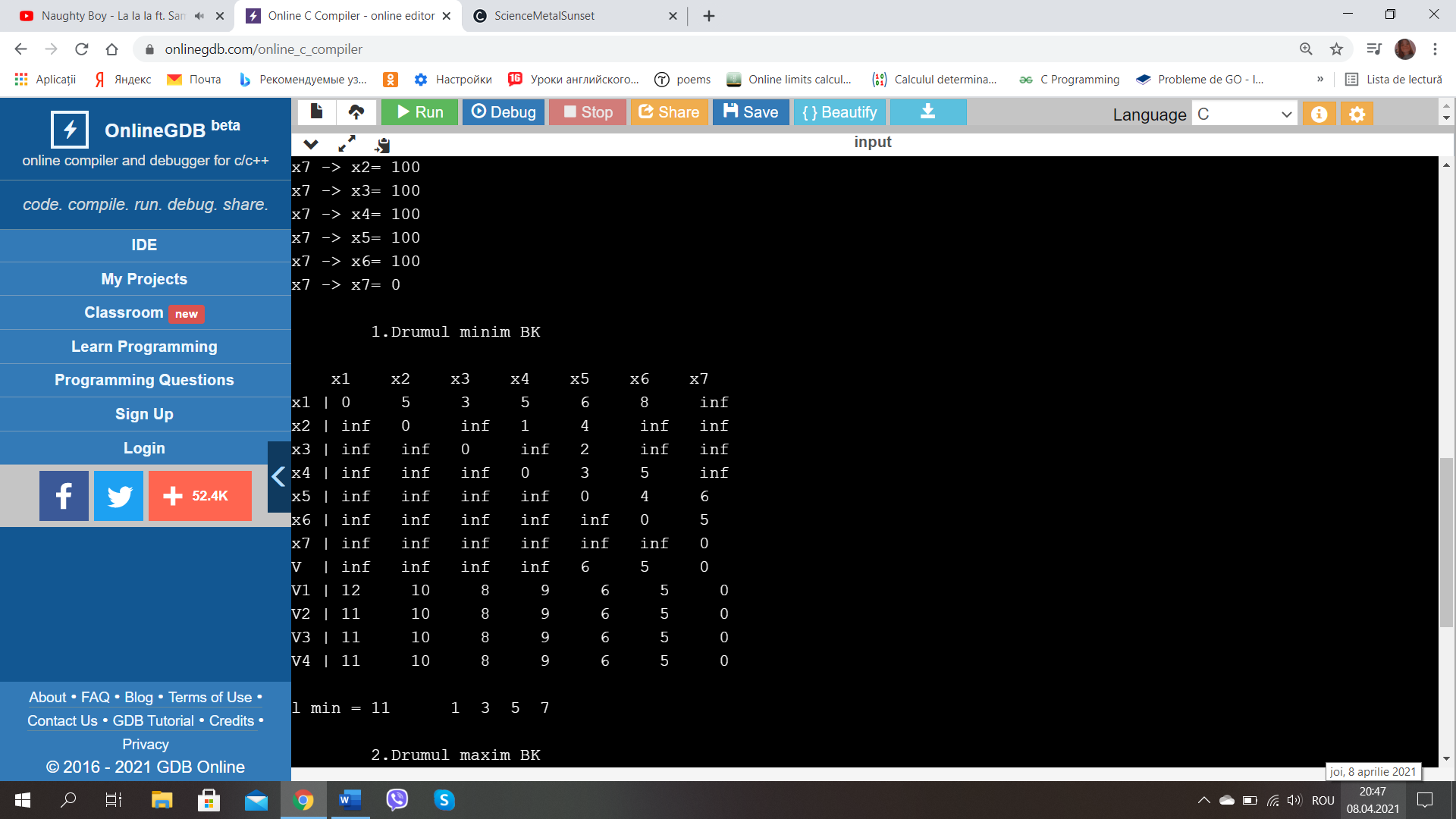
}

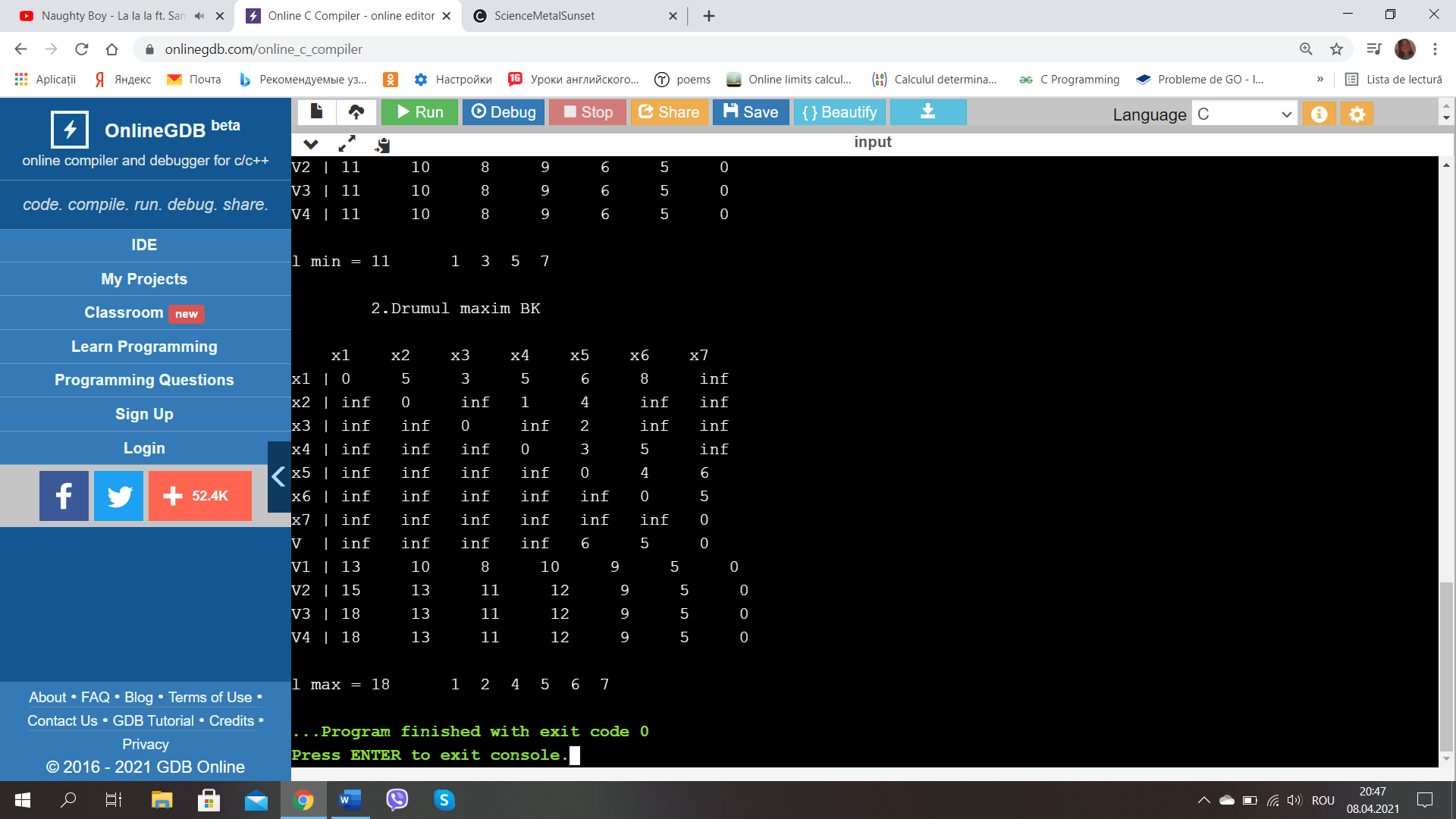
}

**Execuția Programului :**

****

****

****

****

**Concluzia :**

Efectuind aceasta lucrare , neam familiarizat cu algoritmul aflarii drumului minim.Acest algoritm ne permite de a afla drumul minim intre orce doua vurfuri prin metoda lui **Ford** si prin metoda lui **Bellman-Kalaba**.Acest algoritm se aplică pe larg în practică de exemplu la proectarea şoselelor sau a diferitor tipuri de comunicaţii,deci studiind teoretic acum acest algoritm pe viitotr e posibil să-l aplicăm pentru un caz real.