**Programarea calculatoarelor**

**– Seminar nr. 4 –**

**Funcții. Fișiere text. Tablouri**

1. Scrieți o funcție care rearanjează elementele unui tabloului unidimensional cu componente întregi astfel încât toate valorile nule să se afle la sfârșitul tabloului. Ordinea în cadrul secvenţei de elemente nenule nu contează. Scrieţi un program care citeşte de la tastatură două numere naturale și , apoi un şir format din numere întregi cu cel mult 4 cifre fiecare şi, folosind apeluri utile ale funcției definite anterior, verifică dacă şirul conţine cel puţin valori nenule, nu neapărat distincte. În caz afirmativ programul afişează, în fișierul text *date.out*, dintre acestea, iar altfel mesajul NU EXISTA.
2. Scrieţi o funcție care primeşte prin parametrul un număr natural şi returnează suma divizorilor numărului , mai puțin el însuşi. Scrieţi un program care citeşte de la tastatură un număr natural și afișează, folosind apeluri utile ale subprogramului , perechile de numere prietene de forma , cu . Două numere naturale se numesc *numere prietene* dacă suma divizorilor fiecărui număr (mai puțin el însuși) este egală cu celălalt număr.
3. Scrieți o funcție care calculează cifra minimă și cifra maximă pentru un număr natural dat ca parametru. Folosind apeluri utile ale acestei funcții, scrieți un program care afișează toate numerele având toate cifrele egale și sunt mai mici sau egale decât un număr natural citit de la tastatură.
4. Scrieți o funcție care să calculeze valoarea maximă dintr-o secvență cuprinsă între doi indici și () ai unui tablou unidimensional format din numere întregi (elementele tabloului se citesc, printr-o funcție, din fișierul text *numere.in*). Scrieți o funcție care, folosind apeluri utile ale funcției definite anterior, afișează mesajul DA în cazul în care un tablou unidimensional de numere întregi, primit ca parametru, are toate elementele egale sau mesajul NU în caz contrar.
5. Scrieți o funcție care să returneze prima poziție pe care apare într-un tablou unidimensional o valoare având o anumită proprietate sau -1 dacă nu există nicio astfel de valoare. Funcția va avea un parametru de tip pointer la funcție prin care va primi o funcție care implementează criteriul de selecție dorit.
6. Operații cu două numere complexe. Definiti o structura COMPLEX care să permită memorarea părții reale, respectiv a părții imaginare a unui număr complex de forma x+yi. Scrieți funcții care implementează operațiile de adunare, scădere, înmulțire și respectiv împărțire a două numere complexe. Reluați pentru altă pereche de numere de câte ori doreste utilizatorul. *(switch, funcții callback)*
7. Să se citească (cu ajutorul unei funcţii de citire) următoarele informaţii despre elevii participanţi la un concurs de admitere: nume, numărul de înscriere şi cele trei note obţinute. Să se afişeze, printr-o funcţie, informaţiile citite. Să se scrie, în fișierul *lista\_ordonata.txt*, o listă cu elevii participanţi la concurs, ordonaţi alfabetic, notele şi media obţinută (funcţie de ordonare, funcţie de calcul a mediei). Să se afişeze lista elevilor înscrişi la concurs, în ordinea descrescătoare a mediilor.
8. Se dă a, un șir de n numere naturale nenule. Numim o secventa speciala, o secventa a[i], ..., a[j], unde 1<=i<=j<=n, daca cifra de control a sumei secventei respective este 9. Să se determine numărul de secvențe speciale din a.
9. Se citește un numar natural 𝑛 format din o cifră și apoi să realizeze următoarele operații, prin intermediul unor funcții:

a) se construiește în memorie o matrice pătratică de dimensiune 𝑛, având forma indicată mai jos (pentru 𝑛 = 4):

0 1 2 3

1 0 1 2

2 1 0 1

3 2 1 0

b) se scrie matricea, linie cu linie, elementele fiind separate prin spatiu, în fișierul text matrice.txt;

c) se citesc două numere 𝑖 și 𝑗 cuprinse între 0 și 𝑛 − 1 și se interschimbă liniile 𝑖 și 𝑗 din matrice.

Soluții

Ex. 2

#include <stdio.h>

#include <stdlib.h>

int sumad(int x){

int sum=1;

for(int d=2;d<=x/2;d++){

if(x%d==0)sum+=d;

}

return sum;

}

int main(){

int n, ok=0;

scanf("%d",&n);

for(int i=1;i<n;i++)

for(int j=i+1;j<=n;j++){

//int k=sumad(i); int t=sumad(j);

//printf("suma(%d)=%d suma(%d)=%d\n",i, k, j,t);

if(sumad(i)==j && sumad(j)==i)

{

printf("(%d , %d)\n",i, j);

ok=1;

}

}

if (ok==0) printf("Nu exista.");

return 0;

}

Pentru n in [284,1200] scrie: (220, 284)

Ex. 3

#include <stdio.h>

#include <stdlib.h>

void min\_max(int n, int \*mini, int \*maxi) {

\*maxi=0;

\*mini=9;

while (n)

{

if (n % 10 < \*mini) \*mini = n % 10;

if (n % 10 > \*maxi) \*maxi = n % 10;

n /= 10;

}

}

int main()

{

int n;

scanf("%d", &n);

FILE \*g=fopen("date.out","w");

for (int x = 1; x <= n; x++)

{

int mini,maxi;

min\_max(x, &mini, &maxi);

if (maxi == mini)

fprintf(g,"%d ", x);

}

fclose(g);

return 0;

}

Pentru n=889 se scrie, in date.out:

1 2 3 4 5 6 7 8 9 11 22 33 44 55 66 77 88 99 111 222 333 444 555 666 777 888

Ex. 1

#include <stdio.h>

#include <stdlib.h>

int rearanjare (int n, int v[])

{

int cnt=0;

for (int i=1; i<=n-cnt; ++i) {

if (v[i]==0) {

cnt++;

for (int j=i+1; j<=n; ++j) v[j-1]=v[j];

v[n]=0;

--i;

}

}

return cnt;

}

int main()

{

FILE \*f;

f=fopen("date.out","w");

int n,k,v[21],cnt=0;

scanf("%d %d",&n, &k);

for (int i=1; i<=n; ++i) {

scanf("%d",&v[i]);

}

cnt=rearanjare (n,v);

if (n-cnt>=k) for (int i=1; i<=k; ++i) {

/\*char c=v[i]+'0';

fputc(c,f);

fputc(' ',f);\*/

fprintf (f,"%d ",v[i]);

}

else fputs("NU EXISTA",f);

return 0;

}

#include<stdio.h>

int a[101],n,k;

void f(int a[], int n)

{

for(int i = 1; i < n ; i++)

for(int j = i+1 ; j <= n ; j++)

{

if(a[i] < a[j])

{

int aux = a[i];

a[i] = a[j];

a[j] = aux;

}

}

}

int main()

{

scanf("%d %d", &n,&k);

for(int i = 1; i <= n ; i++) scanf("%d", &a[i]);

f(a, n);

int i = 1;

while(a[i] && i <= n){i++;}

FILE \*out = fopen("date.out","w");

if(k < i)

for(int j = 1; j<=k; j++)

fprintf(out, "%d ", a[j]);

else

fprintf(out,"NU EXISTA");

return 0;

}

Ex fara parametri formali. Cum se transmit parametrii unei functii?

#include <stdio.h>

#include <stdlib.h>

int n, v[100];

void citire()

{

FILE \*f = fopen("date.txt","r+");

fscanf(f,"%d",&n);

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

fscanf(f,"%d", &v[i]);

fclose(f);

}

void scrie()

{

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

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

printf("\n");

}

int maxima(int i, int j)

{

int ma=v[i];

for(int k=i+1;k<=j;k++)

if(v[k]>ma)

ma=v[k];

return ma;

}

int main()

{

citire();

scrie ();

printf("%d", maxima(1,5));

return 0;

}

//Ex. 5 seminar 4

#include <stdio.h>

int conditie (int x){

return (x%2 == 0);

}

int cautare (int n, int v[], int(\*f)(int)){

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

if (f(v[i])){

return i;

}

}

return -1;

}

int main()

{

int n, v[100];

scanf ("%d", &n);

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

scanf ("%d", &v[i]);

}

int c = cautare (n, v, conditie);

printf("%d", c);

return 0;

}

//Ex. 1 seminar 4

#include <stdio.h>

#include <stdlib.h>

void rearanjare (int n, int a[100]){

int aux[100], i, j;

for (i=1;i<=n;i++){

aux[i]=a[i];

}

j=0;

for(i=1;i<=n;i++){

if(aux[i]!=0){

j++;

a[j]=aux[i];

}

}

while(j<n){

j++;

a[j]=0;

}

}

int main()

{

int n,k,a[100],i;

scanf("%d %d", &n, &k);

for(i=1;i<=n;i++){

scanf("%d", &a[i]);

}

rearanjare(n,a);

FILE \*output;

output = fopen("date.out", "w");

if(a[k]!=0){

for(i=1;i<=k;i++){

fprintf(output, "%d \n", a[i]);

}

}

else {

fprintf(output, "NU EXISTA \n");

}

fclose(output);

return 0;

}

//Ex. 3 seminar 4

#include <stdio.h>

#include <stdlib.h>

void func(int x, int \*min, int \*max){

\*min = 9; \*max = 0;

if(x == 0)

\*min = 0;

while(x){

if(x%10 > \*max)

\*max = x % 10;

if(x%10 < \*min)

\*min = x % 10;

x /= 10;

}

}

int main()

{

int x, a, b, n;

/\*scanf("%d", &x);

func(x, &a, &b);

printf("%d, %d", a, b);\*/

scanf("%d", &n);

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

func(i, &a, &b);

if(a==b)

printf("%d ", i);

}

return 0;

}

//Ex 9 seminar 4

#include <stdio.h>

int n,a[10][10];

void matrice(int n)

{

int i,j;

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

{

for(j=0;j<n;j++)

{

if(i>j) a[i][j]=i-j;

else a[i][j]=j-i;

}

}

}

void afisare()

{

int i,j;

FILE \*f;

f=fopen("matrice.txt","w");

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

{

for(j=0;j<n;j++) fprintf(f,"%d ",a[i][j]);

fprintf(f,"\n");

}

fclose(f);

}

void interschimbare(int x,int y)

{

int i,aux;

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

{

aux=a[x][i];

a[x][i]=a[y][i];

a[y][i]=aux;

}

}

int main()

{

int x,y,i,j;

scanf("%d",&n);

matrice(n);

afisare();

scanf("%d %d",&x,&y);

interschimbare(x,y);

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

{

for(j=0;j<n;j++) printf("%d ",a[i][j]);

printf("\n");

}

return 0;

}

#include <stdio.h>

int suma1 (int n) {

int s=0;

while(n) {

s += n%10;

n = n/10;

}

return s;

}

int control (int n) {

int s=0;

do {

s=0;

while(n>0) {

s = s + n%10;

n = n/10;

}

n=s;

}while(s>9);

return s;

}

int control1 (int n) {

if(n%9!=0) return n%9;

else return 9;

}

int main()

{

int n, i, j, nr=0, suma=0, control, a[50];

scanf("%d", &n);

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

scanf("%d", &a[i]);

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

{

suma = 0;

for(j=i; j<=n; j++) {

suma = suma + a[j];

if(control1(suma) == 9) {

nr++;

for(int k=i; k<=j; k++)

printf("%d ", a[k]);

printf("\n");

}

}

}

printf("%d", nr);

return 0;

}

//Ex.4 seminar 4 nefinalizat

#include <stdio.h>

int max(int n, int a[100]){

int x = a[0];

for(int i = 1;i<n;i++){

if(a[i]>x){

x=a[i];

}

}

return x;

}

int max1(int i,int j, int a[100]){

int x = a[0];

for(int k = i;k<=j;k++){

if(a[k]>x){

x=a[k];

}

}

return x;

}

int main()

{

int n,a[100],i,j;

scanf("%d", &n);

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

scanf("%d", &a[i]);

}

printf("Valoarea maxima din vector = %d\n",max(n,a));

printf("Valoarea maxima dintre i si j = %d",max1(2,5,a));

return 0;

}

//ex. 1

#include <stdio.h>

#include <stdlib.h>

int v[10000];

void sortt(int n)

{

int aux;

int null = 0 ;

for(int i = 0 ; i < n-1 ; ++i)

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

if(abs(v[i])<abs(v[j]))

{

aux = v[i];

v[i] = v[j];

v[j] = aux ;

}

// Sorteaza descrescator in functie de modulul elementelor

}

int num\_0(int n)

{ int cont = 0 ;

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

{

++cont;

if(!v[i])break;

}

return cont;

// Numara elementele nenule din v

}

int main()

{

int n, k ;

scanf("%d%d",&n,&k);

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

{

int x;

scanf("%i",&x);

v[i] = x;

}

sortt(n);

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

printf("%d ",\*(v+i));

FILE \*f = fopen("date.out","w");

if(k <= num\_0(n))

{

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

fprintf(f,"%d ",\*(v+i));

}

else fprintf(f,"NU EXISTA");

fclose(f);

return 0 ;

}

// ex 1

#include <iostream>

#include <fstream>

using namespace std;

ofstream fout ("date.out");

int v [100];

int knumere (int n )

{

int contor=0;

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

if (v [i] <= 9999 && v[i]!=0)

contor++;

return contor;

}

void sortare (int n )

{

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

if (v [i] == 0)

v [i] += 1234567;

for (int i = 0; i < n - 1; i++)

for (int j = i + 1; j < n; j++)

if (v [i] > v [j])

swap (v [i], v [j]);

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

if (v [i] == 1234567)

v [i] -= 1234567;

}

int main ( )

{

int n,k;

cin >> n>>k;

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

cin >> v [i];

sortare (n );

if (knumere (n) >= k)

{

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

if (v [i] <= 9999 && v[i]!=0)

fout << v [i] << " ";

}

else fout << "NU EXISTA";

return 0;

}

//ex. 2

#include <stdio.h>

#include "file.txt"

int main()

{

int n;

scanf("%d", &n);

for (int i = 1; i < n; i++)

{

for (int j = i + 1; j <= n; j++)

{

if (sumad(i) == j && i == sumad(j))

printf("(%d,%d)\n", i, j);

}

}

return 0;

}

//file.txt contine

int sumad(int x)

{

int s = 0;

for(int i= 1; i< x; i++)

{

if(x % i == 0)

s = s + i;

}

return s;

}

//ex 3

#include <stdio.h>

using namespace std;

void cifre(int x,int\* cmin,int\* cmax)

{

\*cmin=9;

\*cmax=0;

while(x!=0)

{

if(x%10>\*cmax)

\*cmax=x%10;

if(x%10<\*cmin)

\*cmin=x%10;

x=x/10;

}

}

int main()

{

int n,x,i;

scanf("%d",&n);

int c,C;

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

{

cifre(i,&c,&C);

if(c==C)

printf("%d ",i);

}

return 0;

}

//ex

#include <stdio.h>

using namespace std;

void read(int\* n,int v[])

{

FILE \*f=fopen("numere.in","r");

if(f==NULL)

{

printf("Error");

return;

}

fscanf(f,"%d",n);

int i;

for(i=1;i<=\*n;i++)

fscanf(f,"%d",&v[i]);

if(fclose(f)!=0)

{

printf("Error");

return;

}

}

int maxim(int i,int j,int v[])

{

int max=v[i],k;

for(k=i+1;k<=j;k++)

if(v[k]>max)

max=v[k];

return max;

}

int verif1(int n,int v[])

{

int i,ok=1;

for(i=1;(i<n-1)&&ok==1;i++)

if(maxim(i,i+1,v)!=maxim(i+1,i+2,v))

ok=0;

return ok;

}

int verif(int n,int m,int v[])

{

int ok=1,i;

for(i=1;i<=n&&ok==1;i++)

if(v[i]!=m)

ok=0;

return ok;

}

int main()

{

int n,i,v[100];

read(&n,v);

//int m=maxim(1,n,v);

if(verif1(n,v)==1)

printf("DA");

else printf("NU");

return 0;

}

////suma\_maxima

#include <stdlib.h>

#include <stdio.h>

int na, nb, a[100], b[100];

void citire(int \*n, int v[]){

scanf("%d", n);

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

scanf("%d", &v[i]);

}

}

int comparator(const void\* x, const void\* y){

int vx = \*(int\*)x;

int vy = \*(int\*)y;

return vx - vy;

}

int greedy(){

int s = 0, ia = 0, ib = 0;

for(ib = 0; ib < nb && b[ib] <= 0; ++ib, ++ia){

s += a[ia] \* b[ib];

}

for(ib = nb - 1, ia = na - 1; ib >= 0 && b[ib] > 0; --ib, --ia){

s += a[ia] \* b[ib];

}

return s;

}

int main()

{

citire(&na, a);

citire(&nb, b);

qsort(a, na, sizeof(int), comparator);

qsort(b, nb, sizeof(int), comparator);

printf("%d", greedy());

return 0;

}

///

#include <iostream>

#include <fstream>

using namespace std;

ifstream fin("date.in");

ofstream fout("date.out");

int a[10001], lgp[10001],n;

void pd()

{

int i, j, lgmax=0;

for(i=n-1;i>=0;i--)

{

lgp[i] =1;

for(j = i+1; j<=n-1;j++)

if(a[j] >= a[i] && lgp[j]+1 > lgp[i])

lgp[i] = lgp[j]+1;

if(lgp[i] > lgmax)

lgmax = lgp[i];

}

int lc = lgmax, ult=-1;

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

if(lgp[i] == lc && a[i]>=ult)

{

fout<<a[i]<<" ";

ult = a[i];

lc--;

}

}

int main()

{

fin>>n;

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

fin>>a[i];

pd();

return 0;

}

///Scanduri

#include <stdlib.h>

#include <stdio.h>

struct scandura{

int st,dr;

}a[100];

int n;

void citire()

{

scanf("%d",&n);

int i;

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

{

scanf("%d %d",&a[i].st,&a[i].dr);

}

}

int comp(const void \*x, const void \*y)

{

struct scandura vx=\*(struct scandura\*)x;

struct scandura vy=\*(struct scandura\*)y;

return(vx.st-vy.st);

}

void greedy()

{

struct scandura intCrt;

intCrt=a[0];

int i,st,dr,cui=0;

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

{

st=a[i].st;

if(intCrt.dr<a[i].dr)

{

dr=intCrt.dr;

}

else

dr=a[i].dr;

if(st>dr)

{

cui++;

printf("%d ", intCrt.st);

intCrt=a[i];

}

else

{

intCrt.st=st;

intCrt.dr=dr;

}

}

cui++;

printf("%d ", intCrt.st);

}

int main()

{

citire();

qsort(a,n,sizeof(struct scandura),comp);

greedy();

return 0;

}