/\*1. Cum pot afla valorile maxima, respectiv minima pentru tipurile de date char, int; float, double (si precizia).\*/

#include <stdio.h>

#include <limits.h>

#include <float.h>

int main()

{

//tipul char

printf("\nsizeof(char) = %lu \n", sizeof(char));

printf("valoarea minima pt o variabila de tip char este %d \n", CHAR\_MIN);

printf("valoarea maxima pt o variabila de tip char este %d \n\n", CHAR\_MAX);

//tipul int

printf("sizeof(int) = %lu \n", sizeof(int));

printf("valoarea minima pt o variabila de tip int este %d \n", INT\_MIN);

printf("valoarea maxima pt o variabila de tip int este %d \n\n", INT\_MAX);

//tipul float

printf("sizeof(float) = %lu \n", sizeof(float));

printf("valoarea minima > 0 pt o variabila de tip float este %E \n", FLT\_MIN);

printf("valoarea maxima pt o variabila de tip float este %E \n", FLT\_MAX);

printf("valoarea maxima pt o variabila de tip float este %lf \n", FLT\_MAX);

printf("Precizia folosita pentru variabile de tip float este de %d zecimale\n\n\n", FLT\_DIG);

//tipul double

printf("sizeof(double) = %lu \n", sizeof(double));

printf("valoarea minima > 0 pt o variabila de tip double este %E \n", DBL\_MIN);

printf("valoarea maxima pt o variabila de tip double este %E \n", DBL\_MAX);

printf("valoarea maxima pt o variabila de tip double este %lf \n", DBL\_MAX);

printf("Precizia folosita pentru variabile de tip double este de %d zecimale\n\n\n", DBL\_DIG);

return 0;

}

/\* 2. Ordonarea unui vector folosind functia qsort.\*/

#include <stdio.h>

#include <stdlib.h>

int cmp(const void\* a, const void \*b)

{

return \*(int\*)a - \*(int\*)b;

}

int main()

{

int v[] = {0, 5, -6, 9, 7, 12 ,8 ,7, 4};

qsort(v, 9, sizeof(int), cmp);

for( int i = 0; i < sizeof(v)/sizeof(int); i++)

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

printf("\n");

return 0;

}

/\* 3. Un exemplu qsort\*/

#include<stdio.h>

#include<stdlib.h>

#include<limits.h>

int cmp1(const void\* a, const void\* b)

{

int va, vb;

va = \*(int \*)a;

vb = \*(int \*)b;

if(va < vb) return -1;

if(va > vb) return 1;

return 0;

}

int cmp2(const void\* a, const void\* b)

{

return \*(int \*)a - \*(int \*)b;

}

int main()

{

int i;

int v[5] = {0, 5, -6, 9, 7 };

qsort(v,5,sizeof(int),cmp1);

for(i = 0; i< 5; i++)

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

printf("\n");

int w[5] = {0, 5, -6, 9, 7 };

qsort(w,5,sizeof(int),cmp2);

for(i = 0; i < 5; i++)

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

printf("\n");

return 0;

}

/\* 4. Alt exemplu qsort\*/

#include<stdio.h>

#include<stdlib.h>

#include<limits.h>// de ce am nevoie d eaceasta?

int cmp1(const void\* a, const void\* b)

{

int va, vb;

va = \*(int \*)a;

vb = \*(int \*)b;

if(va < vb) return -1;

if(va > vb) return 1;

return 0;

}

int cmp2(const void\* a, const void\* b)

{

return \*(int \*)a - \*(int \*)b;

}

int main()

{

int i;

int v[5] = {INT\_MIN+5, INT\_MIN, INT\_MAX-4, INT\_MAX-1, INT\_MAX};

qsort(v,5,sizeof(int),cmp1);

for(i = 0; i< 5; i++)

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

printf("\n");

int w[5] = {INT\_MIN+5, INT\_MIN, INT\_MAX-4, INT\_MAX-1, INT\_MAX};

qsort(w,5,sizeof(int),cmp2);

for(i = 0; i < 5; i++)

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

printf("\n");

return 0;

}

/\* 5. Inca un exemplu qsort\*/

#include<stdio.h>

#include<stdlib.h>

#include<limits.h>

int cmp1(const void\* a, const void\* b)

{

double va, vb;

va = \*(double \*)a;

vb = \*(double \*)b;

if(va < vb) return -1;

if(va > vb) return 1;

return 0;

}

int cmp2(const void\* a, const void\* b)

{

return \*(double \*)a - \*(double \*)b;

}

int main()

{

int i;

double v[5] = {0.2, 0.1, 0.5, -0.2,-0.3};

qsort(v,5,sizeof(double),cmp1);

for(i = 0; i< 5; i++)

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

printf("\n");

double w[5] = {0.2, 0.1, 0.5, -0.2,-0.3};

qsort(w,5,sizeof(double),cmp2);

for(i = 0; i < 5; i++)

printf("%f ", w[i]);

printf("\n");

return 0;

}

/\* 6. Si inca un exemplu qsort\*/

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int cmpSiruri(const void\* a, const void\* b)

{

if(strlen(\*(char\*\* )a) < strlen(\*(char \*\*)b)) return -1;

if(strlen(\*(char\*\* )a) > strlen(\*(char \*\*)b)) return 1;

return 0;

}

int main()

{

char\* nume[5] = {"Miruna","Claudiu","Ionut","Paul","Stefan"};

int i;

for(i=0;i<5;i++)

printf("%p %p %s \n",nume+i, nume[i], nume[i]);

qsort(nume,5,sizeof(char\*), cmpSiruri);

printf("Dupa qsort \n");

for(i=0;i<5;i++)

printf("%p %p %s \n",nume+i, nume[i], nume[i]);

return 0;

}

/\* 7. Sa se determine cate linii are un fisier text dat.\*/

#include<stdio.h>

#include<stdlib.h>

int main()

{

FILE \*f;

char linie[1000];

int numarLinii = 0;

f = fopen("test.txt","r");

if (f == NULL)

{

printf("Eroare la deschiderea fisierului");

exit(0);

}

while (fgets(linie,1000,f) != NULL)

numarLinii++;

printf("Fisierul are %d linii\n",numarLinii);

fclose(f);

return 0;

}

/\* 8. Exemplu de citire-scriere cu format (de la tastatura in fisier, din fisier pe ecran)\*/

#include<stdio.h>

int main()

{

FILE \* f;

int x;

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

if (f==NULL)

{

fprintf(stdout,"Nu pot deschide fisiserul! \n");

return 0;

}

while(fscanf(stdin,"%d",&x))

fprintf(f,"%d \n",x);

fclose(f);

f = fopen("numere.txt", "r");

while(fscanf(f,"%d",&x) != EOF)

fprintf(stdout,"%d \n",x);

return 0;

}

/\* 9. Exemplu de copiere a continutului unui fisier in alt fisier.\*/

#include<stdio.h>

#include <stdlib.h>

int main()

{

FILE \*fin, \*fout;

char \*numeFisier = "exemplu.txt";

fin = fopen(numeFisier,"r");

fout = fopen("exemplu\_copie.txt","w");

if ((fin == NULL) || (fout == NULL))

{

printf("Eroare la deschiderea fisierelor");

return 0;

}

printf("Incepem copierea fisierului\n");

char c;

while ((c=fgetc(fin)) != EOF)

fputc(c,fout);

printf("Am terminat copierea fisierului\n");

if ((fclose(fin) != 0) || (fclose(fout) != 0))

{

printf("Eroare la inchiderea fisierelor");

return 0;

}

return 0;

}

/\* 10. Citirea caracterelor dintr-un fisier (se scrie codul ASCII si caracterul).\*/

#include<stdio.h>

int main()

{

FILE \*f;

int c;

f = fopen("exemplu.txt", "r");

if (f==NULL)

{

fprintf(stdout,"Nu pot deschide fisierul! \n");

return 0;

}

printf("Incepem citirea din fisier \n");

while (!feof(f))

{

c = fgetc(f);

printf("%d %c\n",c,c);

}

printf("\n");

fclose(f);

return 0;

}

/\* 11. Adrese de pointeri.\*/

#include <stdio.h>

int main()

{

int v[5];

int \*p;

p = &v[0];

printf("Adresa lui v[0] este %p \n", p);

printf("Valoarea lui v este %p \n",v);

printf("Adresa lui v este %p \n",&v);

if (p==v)

printf("Aceeasi adresa \n");

printf(" %p \n %p \n %p \n %p \n %p \n", v,v+1,v+2,v+3,v+4);

printf("\n %p \n %p \n %p \n %p \n %p \n", v,v-1,v-2,v-3,v-4);

p =p + 2;

printf("Adresa spre care pointeaza acum p este %p \n",p);

return 0;

}

/\* 12. Determinarea valorile minima si maxima dintr-un vector de elemente intregi folosind structuri (3 exemple de functii).\*/

#include <stdio.h>

typedef struct

{

int min;

int max;

} minmax;

minmax calculeazaMinimMaximTablou1(int v[], int n)

{

int minim = v[0];

int maxim = v[0];

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

{

if (minim > v[i])

minim = v[i];

if (maxim < v[i])

maxim = v[i];

}

minmax x;

x.min = minim;

x.max = maxim;

return x;

}

void calculeazaMinimMaximTablou2(int v[],int n, int\* min,int \*max)

{

\*min = v[0];

\*max = v[0];

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

{

if (\*min > v[i])

\*min = v[i];

if (\*max < v[i])

\*max = v[i];

}

}

int calculeazaMinimMaximTablou3(int v[],int n, int\* min)

{

\*min = v[0];

int max = v[0];

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

{

if (\*min > v[i])

\*min = v[i];

if (max < v[i])

max = v[i];

}

return max;

}

int main()

{

int v[] = {1, 4, 5, 7, 6, 8, 2, 3};

minmax a = calculeazaMinimMaximTablou1(v,8);

printf("a.min = %d \t a.max = %d \n",a.min,a.max);

int i,j;

calculeazaMinimMaximTablou2(v,8, &i,&j);

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

j = calculeazaMinimMaximTablou3(v,8, &i);

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

}

/\* 13. ”Expresii”!\*/

#include <stdio.h>

int suma(int n, int (\*expresie)(int))

{

int i, s = 0;

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

s = s + expresie(i);

return s;

}

int expresie1(int x)

{

return x;

}

int expresie2(int x)

{

return x\*x;

}

int main()

{

int S1 = suma(5,expresie1);

printf("S1 = %d\n",S1);

int S2 = suma(5,expresie2);

printf("S2 = %d\n",S2);

return 0;

}

/\* 14. Exemplu pointeri la functii\*/

#include <stdio.h>

int suma(int a, int b)

{

return a + b;

}

int diferenta(int a, int b)

{

return a - b;

}

int main()

{

int (\*pf)(int,int);

pf = &suma;

int s = (\*pf)(2,5);

printf("s = %d \n",s);

pf = diferenta;

int d = pf(2,5);

printf("d = %d \n",d);

return 0;

}

/\* 15. Exemplu de transmitere a valorilor\*/

#include <stdio.h>

int f1(int a, int b)

{

a++;

b++;

printf("In f1 avem a= %d \t b = %d \n",a,b);

return a + b;

}

int f2(int \*a, int b)

{

\*a = \*a + 1;

b++;

return \*a + b;

}

int main()

{

int x = 5, y = 8;

int z = f1(x, y);

printf("In main dupa f1 avem x = %d, y = %d, z = %d \n",x,y,z);

z = f2(&x, y);

printf("In main dupa f1 avem x = %d, y = %d, z = %d \n",x,y,z);

return 0;

}

/\* 16. Alocare dinamica. Citire + afisare tablou\*/

#include <stdio.h>

#include <stdlib.h>

int citire(int\*\* v)

{

int i,n;

printf("n=");scanf("%d",&n);

\*v = (int \*) malloc(n\*sizeof(int));

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

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

return n;

}

int main()

{

int n, i, \*p = NULL;

n = citire(&p);

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

printf("p[%d] = %d\n",i,p[i]);

return 0;

}

/\* 17. Accesul la elementele unui tablou\*/

#include <stdio.h>

int main()

{

int v[5] = {0, 2, 4, 10, 20};

int \*p = v;

int i;

for(i=0; i<5; i++)

{

printf("Accesam elementul %d din vectorul v prin intermediul lui p.\n",i);

printf("Valoarea acestui element este = %d \n", \*(p+i));

printf("Valoarea elementului poate fi accesata si prin p[%d] = %d \n\n",i, p[i]);

}

return 0;

}

/\* 18. Transmiterea unui tablou unidimensional ca parametru al unei functii\*/

#include <stdio.h>

void afiseazaTablou(int v[])

{

int i;

for(i=0;i<sizeof(v)/sizeof(int);i++)

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

printf("Dimensiunea lui v este %lu \n", sizeof(v));

}

int main()

{

int v[5] = {1,3,5,7,9};

afiseazaTablou(v);

return 0;

}

/\* 19. Alocarea si realocarea de memorie pentru un tablou de intregi\*/

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \*a, \*aux;

a = (int \*) malloc(100 \* sizeof(int));

if(!a)

{

printf("Nu pot aloca memorie");

return 1;

}

aux = (int \*) realloc(a,200 \* sizeof(int));

if(!aux)

{

printf("Nu pot dimensiona blocul a");

free(a);

return 1;

}

printf("Redimensionare reusita \n");

a = aux;

free(a);

return 0;

}

/\* 20. Pointer dublu\*/

#include<stdio.h>

int main()

{

int a = 5, \*p = &a, \*\*q = &p;

printf("Valoarea lui a poate fi obtinuta astfel:\n");

printf("Direct: a = %d \n",a);

printf("Prin p: \*p = %d \n", \*p);

printf("Prin q: \*\*q = %d \n", \*\*q);

printf("Adresa lui a este: %p \n", &a);

printf("Adresa lui p este: %p \n", &p);

printf("Adresa lui q este: %p \n", &q);

printf("Adresa spre care pointeaza p este: %p \n", p);

printf("Adresa spre care pointeaza q este: %p \n", q);

printf("Adresa spre care pointeaza p este: %p \n", \*q);

return 0;

}

/\* 21. Tablou bidimensional\*/

#include <stdio.h>

#include <stdlib.h>

int main() {

int L, C, i, j;

int \*\*p; // Adresa matrice

printf("Nr de linii L = "); scanf("%d", &L);

printf("Nr de coloane C = "); scanf("%d", &C);

p = (int\*\*) malloc(L \* sizeof(int\*));

printf("Sizeof(int\*) = %lu \n",sizeof(int\*));

printf("Pointerul p contine adresa %p \n",p);

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

{

p[i] = calloc(C, sizeof(int));

printf("Linia %d incepe la %p \n",i,p[i]);

}

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

for (j = 0; j < C; j++) {

p[i][j] = L \* i + j + 1;

printf("Adresa lui p[%d][%d] este = %p \n",i,j,&p[i][j]);

}

}

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

for (j = 0;j < C; j++) {

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

}

printf("\n");

}

return 0;

}

/\* 22. Acces la elementele unui tablou bidimensional\*/

#include <stdio.h>

int main()

{

int a[5][5],i,j;

for(i = 0; i < 5; i++)

for(j = 0; j < 5; j++)

a[i][j] = i\*j;

i = 3;

for (j=0;j<5;j++)

{

printf("Valoarea lui a[%d][%d] este %d \n",i, j, a[i][j]);

printf("Valoarea lui a[%d][%d] este %d \n",i, j, \*(\*(a+i)+j));

}

return 0;

}

/\* 23. Transmiterea (statica) a unui tablou 2d ca parametru al unei functii. Un exemplu\*/

#include <stdio.h>

void printeazaMatrice(int a[][5], int l, int c)

{

int i,j;

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

{

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

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

printf("\n");

}

return;

}

int main()

{

int a[5][5],i,j;

for(i = 0; i < 5; i++)

for(j = 0; j < 5; j++)

a[i][j] = i\*j;

printeazaMatrice(a,5,5);

return 0;

}

/\* 23. bis. Transmiterea (statica) a unui tablou 2d ca parametru al unei functii. Alt (acelasi!) exemplu\*/

#include <stdio.h>

#include <stdlib.h>

void afiseazaMatrice(int x[][4], int nrLinii, int nrColoane)

{

int i,j;

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

for (j = 0;j < nrColoane; j++) {

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

}

printf("\n");

}

}

int main() {

int i ,j, L = 3, C = 4;

int p[3][4];

for (i = 0; i < 3; i++) {

for (j = 0; j < 4; j++) {

p[i][j] = L \* i + j + 1;

}

}

afiseazaMatrice(p,L,C);

return 0;

}

/\* 24. Transmiterea dinamica a unui tablou 2d ca parametru al unei functii (2 exemple)\*/

#include <stdio.h>

#include <stdlib.h>

void afiseazaMatrice(int \*\*x, int nrLinii, int nrColoane)

{

int i,j;

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

for (j = 0;j < nrColoane; j++) {

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

}

printf("\n");

}

}

void afiseazaMatrice1(int\* x[], int nrLinii, int nrColoane)

{

int i,j;

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

for (j = 0;j < nrColoane; j++) {

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

}

printf("\n");

}

}

int main() {

int L, C, i, j;

int \*\*p; // Adresa matrice

L = 3, C = 4;

p = (int\*\*) malloc(L \* sizeof(int\*));

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

{

p[i] = calloc(C, sizeof(int));

printf("Linia %d incepe la %p \n",i,p[i]);

}

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

for (j = 0; j < C; j++) {

p[i][j] = L \* i + j + 1;

printf("Adresa lui p[%d][%d] este = %p \n",i,j,&p[i][j]);

}

}

afiseazaMatrice(p,L,C);

afiseazaMatrice1(p,L,C);

return 0;

}

/\* 25. Pointeri generici. Ex\*/

#include<stdio.h>

int main()

{

int i = 6;

void \*p;

p = &i;

printf("p pointeaza catre un int cu valoare = %d\n", \*p);

return 0;

}

/\* 26. Pointeri generici. Alt ex\*/

#include<stdio.h>

int main()

{

int i = 6;

char c = 'a';

void \*p;

p = &i;

printf("p pointeaza catre un int cu valoarea = %d\n", \*(int\*) p);

p = &c;

printf("p pointeaza catre un char cu valoarea = %c\n", \*(char\*) p);

return 0;

}