**Pointeri. Alocare dinamica. Fisiere text. Functii**

**I. Repere teoretice**

**A. Pointeri**

**1. Structura generala a memoriei**

* Zone în memoria internă alocate unui program: segmentul de date, segmentul de stiva (pentru memorarea parametrilor, variabilelor locale)
* Unitatea independenta de adresare: octetul, identificat prin numarul sau de ordine = adresa sa;
* Adresa unei variabile: adresa la care se afla octetul sau cel mai putin semnificativ.

Ex: int a=16909125;

// a=0x01020345; in baza 16

// 0000.0001.0000.0010.0000.0011.0100.0101

**2. Operatori**

* Operatorul de adresare (referentiere): & (Expresia &varare ca valoare adresa variabilei var**)**
* Operatorul de dereferentiere: \* (Expresia \*adr are ca valoare continutul adresei memorate în variabila adr**)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Adrese  ⇓ | Continutul  memoriei  ⇓ |  |
|  | DS:0xFFG2 |  |  |
| Adresa variabilei **a:**⇒ **&a** | DS:0xFFG3 | 01000101 | ⇐ Spatiul de memorie  alocat variabilei a |
|  | DS:0xFFG4 | 00000011 |
|  | DS:0xFFG5 | 00000010 |
|  | DS:0xFFG6 | 00000001 |
|  | DS:0xFFG7 |  |  |

**3. Tipuri de date pentru adresarea memoriei - pointeri**

**Valori** – adrese de memorie (exemple: DS:0xFFG2; 0xFFG2; 0; NULL); Observatie : valorile de tip pointer nu pot fi citite

**Memorare:** ocupa 4 octeti de memorie

**Tip**: pointer de un anumit tip, precizat la declarare

**Rol**: adreseaza (= pointeaza = poate accesa) zona de memorie identificata prin valoarea sa si a carei dimensiune depinde de tipul precizat

**4. Declarare pointeri**

**a. pointeri catre tipuri simple**

* fara initializare: T \*p; sau T\* p; sau T \* p; p este o variabila de tip pointer catre T și reține adresa unei zone de memorie in care este retinuta o valoare de tip T(accesul la această valoare: \*p este o dată de tip T)
* cu initializare: T \*p=&var; p este o variabila de tip pointer catre T și retine adresa variabilei var, care trebuie sa fie de tip T
* lista de pointeri: T \*p1, \*p2;
* pointeri la pointeri: T \*\*p;

**b. pointeri catre structuri**

struct S {

T1 camp1;

T2 camp2;

….

} \*p;

p este pointer de tip structura și retine adresa unei date de tip structura (accesul la această valoare: \*p este o data de tip structura; accesul la un camp al structurii: (\*p).camp sau p->camp)

**c. pointeri si tablouri alocate static**

T p[dim]; //tablou cu dim elemente de tip T (p reprezinta adresa primului element)

T \*p[dim]; //tablou de dim pointeri

T (\*p)[dim]; //pointer la un tablou cu dim elemente de tip T

T p[dim1][dim2]; //matrice de elemente tip T

T \*p[dim1][dim2]; //matrice de pointeri

T (\*p)[d1][d2]; //pointer catre o matrice cu d1 linii si d2 coloane, cu elemente de tip T

**5. Operatii specifice cu pointeri**

* adunarea unui intreg (pozitiv sau negativ): p+n (se obtine adresa p+n\*(sizeof(T)))
* scaderea a doi pointeri (de acelasi tip) - rezultatul este un numar intreg (Observatie: doi pointeri nu se pot aduna)
* atribuiri – între pointeri de același tip

**6. Alocare dinamică**

T\*p;

p=(T\*)malloc(n\*sizeof(T)); //alocarea memoriei pentru n elemente de tip T

free(p); //eliberarea memoriei alocate

**B. Sistemul de intrare/ieșire**

**Sistemul** nu face parte din limbaj, ci este adăugat printr-un set de funcții din biblioteca standard

**1. Stream-uri** – dispozitive logice prin care se face transferul

* **Tipuri:** text (secvențe de caractere organizate în linii), binar (secvențe de octeți fără nicio structură)

**2. Dispozitive logice predefinite:** se creează automat la lansarea în execuție a programului și sunt inchise automat la incheierea executiei;

* Tipuri: text (stdin, stdout, stderr), binar – specifice DOS (stdprn, stdaux)
* Funcții de intrare/ieșire pentru consolă (utilizează implicit stdin, stdout, stderr):
* Operații la nivel de caracter: getchar, getche, getch, putchar;
* Operații cu șiruri de caractere: gets, puts
* Operații de citire/scriere cu formatare: printf, scanf

**3. Operații cu stream-uri:**

* **inițierea transferului (deschiderea fluxului)** – se creează un stream și se asociază unui dispozitiv logic DOS sau unui fișier de pe disc (se creeaza o variabila de tip FILE si eventual se aloca o zona tampon prin intermediul careia se fac transferurile)

**FILE \* f;**

**f=fopen(numeFisier,mod);**

mod: [r/w/a][+][t/b]

* **incheierea transferului**  **(inchiderea fluxului)** – se elibereaza memoria alocata la deschidere, dupa golirea zonei tampon

**fclose(f);**

* Funcții de intrare/ieșire cu caracter general:
* Operații de testare: ferr/clearerr, feof
* Operații la nivel de caracter/octet/cuvant: getc, putc, getw, putw;
* Operații cu șiruri de caractere: fgets, fputs
* Operații de citire/scriere cu blocuri de octeti: fread, fwrite
* Operații de citire/scriere cu formatare: fscanf, fprintf
* Operații pentru indicatorul de pozitie: ftell, fgetpos, fsetpos, fseek, rewind
* Operații generale cu fișiere: remove, rename

**C. Funcții**

**1. Definirea funcției**

* Prototipul funcției
* Parametri formali – precizați la definirea funcției: de intrare (transmiși prin valoare), de ieșire (transmiși prin adresa), de intrare-ieșire (transmiși prin adresa)

**2. Apelul funcției**

* Parametri efectivi/actuali

**Functii cu numar variabil de parametri**

**Reguli de proiectare a subprogramelor cu numar variabil de parametri**

* Parametri prezenti constant (parametrii ficsi ofera informatii despre ceilalti)
* Parametri care urmeaza optional: …

**II. Exercitii**

**A. Exerciții cu pointeri**

|  |  |
| --- | --- |
| **Item** | **Raspuns asteptat** |
| Fie declararea  int i=3, \*p=&i; | |
| Cata memorie se aloca? | Se aloca – 4 octeti pentru variabila i și 4 octeti pentru variabila p |
| Care este rolul lui p? | Pointeaza zona de memorie alocata lui i (cei 4 octeti) |
| Care este valoarea variabilei p? | Adresa variabilei intregi i |
| Ce valoare are j daca  int j;  j=\*p+1 | 4 |
| Fie declararea  long i=10, \*p=&i; | |
| Cata memorie se aloca? | Se aloca – 4 octeti pentru variabila i și 4 octeti pentru variabila p |
| Care este rolul lui p? | Pointeaza zona de memorie alocata lui i (cei 4 octeti) |
| Fie declararea  long i=10, \*p=&i; | |
| Ce reprezeinta &p? | Adresa la care este memorat p |
| Ce se afla la adresa &p? | O valoare de tip adresa |
| Ce se declara in  int \*a,b; | O variabila de tip pointer catre un intreg – a  O variabila de tip intreg – b |
| Fie declararea  int i=3, \*p=&i, \*\*q=&p; | |
| Cata memorie i se aloca lui q? | 4 octeti |
| ce afiseaza  printf(“%p”,q); | Adresa la care se memoreaza p |
| ce afiseaza  printf(„%p”,\*q); | Continutul adresei, deci valoarea lui p |
| ce afiseaza  printf(„%d”,\*\*q); | Valoarea aflata la adresa p, adica 3 |

**Ex.2.**

|  |  |
| --- | --- |
| **Item** | **Raspuns asteptat** |
| struct elev{  char nume[20];  float medie[10];  } \*p; | |
| Cata memorie se aloca? | Se aloca - 4 octeti pentru variabila p |
| Care este valoarea variabilei p? | Adresa unei date de tip structura |
| Intr-o expresie, de ce tip este  \*p | Structura |
| (\*p).nume | Sir de caractere |
| p->medie[0] | Numar real |

**Ex3.**

|  |  |
| --- | --- |
| **Item** | **Raspuns asteptat** |
| Fie declararea  int \*p[10]; | |
| Ce este p? | Un tablou care poate retine 10 pointeri, fiecare fiind de tip pointer catre un intreg |
| Cata memorie se aloca? | 4x10 |
| Ce tip de valoare are p[1]? | Adresa unui numar intreg |
| Fie declararea  int (\*p)[10]; | |
| Ce este p? | Un pointer catre un tablou de numere intregi |
| Cata memorie se aloca? | 4 octeti |

**Ex. 4.**

|  |  |  |
| --- | --- | --- |
| **Item** | **Raspuns asteptat** | |
| Fie declararea  int a,\*p=&a, \*q;  q=p+2; | | |
| Daca p are valoarea  0xFF00, ce valoare are q? | 0xFF08 (s-au adaugat 2x4 octeti) | |
| Fie declararea  long long a, \*p=&a, \*q;  q=p+2; | | |
| Daca p are valoarea 0xFF00, ce valoare are q? | 0xFF10 (s-au adaugat 2x8 octeti = 16(10)=10(16)) | |
| Fie declararea  int p[10];  //  int \*p;  p=(int \*)malloc(10\*sizeof(int)); | | |
| Ce reprezinta p+i? | Adresa elementului de pe pozitia i | |
| Ce reprezinta  \*(p+i) | p[i] | |
| Ce reprezinta  \*(i+p) | p[i] | |
| Fie declararea  int p[2][3];  //  int \*\*p;  p=(int \*\*)malloc(2\*sizeof(int \*));  for(i=0;i<2;i++)  p[i]=(int \*)malloc(3\*sizeof(int)); | | |
| Ce reprezinta p? | Vector de 2 elemente, fiecare element fiind un vector de 3 elem. | |
| Ce reprezinta p+i? | Adresa liniei i a matricei (al i-lea vector) | |
| Ce reprezinta  \*(\*(p+i)+j) | p[i][j] | |
| Fie declararea  int p[9][5][3];  //  int \*\*\*p;  p=(int \*\*\*)malloc(9\*sizeof(int \*\*));  for(i=0;i<9;i++)  {  p[i]=(int \*\*)malloc(5\*sizeof(int \*));  for(j=0;j<5;j++)  p[i][j]=(int \*)malloc(3\*sizeof(int));  } | | |
| Ce reprezinta p? | | Vector de 9 elemente, fiecare element fiind o matrice de 5 linii si 3 coloane (vector de 9 elemente, fiecare fiind un vector de 5 elemente, fiecare fiind un vector de 3 elemente) |
| Ce reprezinta  \*p, \*(p+1), \*(p+2).../  p[0], p[1], p[2].... | | Primele trei matrice dintre cele 9 |
| Ce reprezinta  \*(p+i), \*(p+i)+1, \*(p+i)+2.../  p[i], p[i]+1, p[i]+2 .... | | Primele 3 linii din matricea p[i] |
| Ce reprezinta  \*(\*(p+i)+j), \*(\*(p+i)+j)+1, \*(\*(p+i)+j)+2.../  p[i][j], p[i][j]+1, p[i][j]+2 .... | | Primele 3 elemente din linia j a matricei p[i] (primele trei elemente ale vectorului p[i][j] |
| Ce reprezinta  \*(\*(\*(p+i)+j)+k) /  \*(p[i][j]+k)/  p[i][j][k] | | Elementul k al vectorului j din matricea i |

**Ex. 5.**

|  |  |
| --- | --- |
| **Item** | **Raspuns asteptat** |
| Fie declararile  T1 a, b, \*p1=&a, \*p2=&b;  T2 c,d,\*q1=&c, \*q2=&d;  Caredintre atribuirile de mai jos sunt corecte? | |
| p1=p2; |  |
| p1=q1; | //incorect |
| \*p1=\*q1; | //incorect (corect daca se poate face transformarea tipurilor) |
| \*p1=\*p2; |  |
| p1=&a; |  |
| p1=&c; | //incorect |
| p1=&p2; | //incorect |

**B. Exercitii cu fisiere**

|  |  |  |
| --- | --- | --- |
| 1. | Studiați efectul programului de mai jos: |  |
|  | #include<stdio.h>  #include<conio.h>  #include<stdlib.h>  int main()  {  FILE \*out;  char c;  if ((out=fopen(“date.out”, "w"))==NULL)  {  printf("Nu se poate crea fisierul.\n");  return 1;  }  puts("Scrieti textul incheiat cu #:");  do {  c=getchar();  if(c!='#')  putc(c,out);  } while (c !='#');  fclose(out);  return 0;  } | Creeaza fisierul date.out, apoi introduce in acesta textul citit de la tastatura, pana la #. |
| 2. | Studiați efectul programului de mai jos: |  |
|  | #include<stdio.h>  #include<conio.h>  #include<stdlib.h>  int main()  {  FILE \*in;  char c;  in = fopen(”date.in”, "r");  if(in == NULL)  {  printf ("Nu se poate deschide fisierul precizat!");  return 1;  }  do {  c=fgetc(in);  if(!feof(in) && c!='#')  putchar(c);  }while(!feof(in) && c!='#');  fclose(in);  return 0;  } | Citeste textul din fisierul cu numele date.in, (pana la intalnirea lui #) apoi il scrie pe ecran. |

**C. Exerciții cu funcții**

|  |  |  |
| --- | --- | --- |
| 1. | Fie programul: | |
|  | int a,b,c;  void P(int a, int b)  {  a=b+1;  b\*=2;  c=a+b+c;  }  int main()  {  a=10; b=100;c=1000;  P(a,b);  printf("%d %d %d",a,b,c);  return 0;  } | |
|  | Ce se afiseaza? | 10 100 1301 |
|  | Ce se afiseaza daca se inlocuieste apelul P(a,b) cu P(b,a)? | 10 100 1301 |
|  | Ce se afiseaza daca se inlocuieste apelul P(a,b) cu P(a,c)? | 10 100 4001 |
|  | Ce se afiseaza daca se inlocuieste apelul P(a,b) cu P(c,c)? | 10 100 4001 |
|  | int a,b,c;  void P(int a, int \*b)  {  a=\*b+1;  (\*b)\*=2;  c=a+\*b+c;  }  int main()  {  a=10; b=100;c=1000;  P(a,&b);  printf("%d %d %d",a,b,c);  return 0;  } | |
|  | Ce se afiseaza? | 10 200 1301 |
|  | Ce se afiseaza daca se inlocuieste apelul P(a,&b) cu P(b,&a)? | 20 100 1031 |
|  | Ce se afiseaza daca se inlocuieste apelul P(a,&b) cu P(a,&c)? | 10 100 5001 |
|  | Ce se afiseaza daca se inlocuieste apelul P(a,&b) cu P(c,&c)? | 10 100 5001 |

|  |  |  |
| --- | --- | --- |
| 2. | Studiați efectul programului de mai jos. | |
|  | int suma(int n,...)  {  int \*adr=&n+1;  int s=0,i;  for(i=0;i<n;i++)  { s+=\*adr;  adr++;  }  return s;  }  int main ( ) {  printf(„%d”,suma(3,9,8,3);  printf(„%d”,suma(4,9,8,5,3);  return 0;  } |  |

**III. Aplicații:**

**<pointeri>**

1. Afișați conținutul, în baza 16, al celui mai puțin semnificativ octet al unei variabile a de tip int.

Indicație: se interpretează conținutul ca fiind al unei variabile de tip char: \*(char\*)&a

2. Afișați conținutul, în baza 16, al celui mai semnificativ octet al unei variabile de tip long long.

**<fișiere text>**

3.Fisierul text date.in contine cuvinte si numere intregi, scrise pe mai multe linii. Pe fiecare linie exista cel putin o entitate lexicala (cuvinte si numere), iar entitatile lexicale aflate pe aceeasi linie sunt separate prin cate un spatiu. Scrieti un program care afiseaza pe ecran:

a) numărul de cifre din fișier;

b) numărul de entitati lexicale din fișier;

c) numărul de linii din fișier;

d) numarul de entitati lexicale aflate pe fiecare linie.

4. Fisierele A.txt si B.txt contin numere naturale ordonate crescator.

a) Scrieți un program care verifică dacă cele două fișiere sunt identice.

b) Scrieti un program care sa creeze un fisier C.txt care sa contina numerele din cele doua fisiere precizate mai sus, ordonate crescator.

5. Eliminați toate spatiile din fisierul date.in.

**<funcții>**

**6. Numere complexe**

a) Scrieți funcții pentru operatii cu numere complexe:

* citirea unui numar complex;
* afisarea unui număr complex;
* adunarea a două numere complexe;
* inmultirea a două numere complexe;
* impartirea a doua numere complexe.

b) Sa se citeasca apoi N numere complexe. Utilizand funcțiile create anterior, calculați

- suma celor N numere

- produsul celor N numere

Exemple: pentru N=3 si numerele

1 2

3 3

4 2

se afișează

Suma: 8+7i

Produsul: -30+30i

**7. Fractii**

a) Scrieți câte o funcție care:

* sa calculeze cmmdc a doua numere primite ca parametri
* sa aduca la forma ireductibila o fractie data prin perechea numarator, numitor
* aduna doua fractii

b) Utilizand subprogramele definite anterior scrie un program care sa calculeze suma a N fractii, rezultatul fiind o fractie ireductibila data prin perechea numarator, numitor

Datele se citesc astfel: pe prima linie N, iar pe urmatoarele N linii perechi de forma a b (numarator, numitor)

**8. Numere speciale**

a) Scrie funcții care:

* calculeaza suma divizorilor unui numar
* calculeaza oglinditul unui numar (nr. Obtinut prin inversarea ordinii cifrelor sale) ex. Pentru N=123, oglinditul este 321
* verifica daca un numar este prim
* verifica daca un numar este perfect (este egal cu suma divizorilor sai) ex. 6 este perfect (=1+2+3)
* verifica daca doua numere sunt prietene (fiecare este egal cu suma divizorilor celuilalt) ex. 220 si 284

b) Utilizand funcțiile definite anterior scrieți urmatoarele programe:

P1. Sa se genereze toate perechile de numere prime formate din trei cifre cu proprietatea ca fiecare este oglinditul celuilalt

(101 ,101); (107, 701); (113, 311)….

P2. Se da un sir de numere naturale. Sa se afiseze sirul obtinut prin eliminarea tuturor numerelor perfecte precum si a tuturor perechilor de numere prietene aflate pe pozitii consecutive.

Ex. daca sirul este 11, 28, 6, 133, 220, 284, 7, 99

se obtine 11, 133, 7, 99