Programare Procedurala
Laborator 8

Pointeri la structuri

Putem defini pointeri la structuri in acelasi mod in care am defini un pointer la orice alta variabila.

Declarare:
```
struct struct_name *struct_pointer;
```

Pentru a accesa membrii unei structuri prin intermediul pointerilor, folosim operatorul ->
```
struct_pointer->member_name;
```

Instructiunea de mai sus este echivalenta cu:
```
(*struct_pointer).member_name;
```

Pentru a trimite un pointer la o structura ca argument unei functii, folosim urmatoarea sintaxa:
```
return_type function_name(struct struct_name *param_name);
```

Exemplu:

```
struct Book {
     int book_id;
     char title[50];
     char author[50];
     char subject[100];
};
void printBook(struct Book *book);

int main () {
     struct Book *book_pointer, book1;
     book_pointer = &book1;
     //specificatia pentru book1:
     book1.book_id = 2000;
     strcpy(book1.title, "Defence Against the Dark Arts");
     strcpy(book1.author, "Severus Snape");
     strcpy(book1.subject, "Defence Against the Dark Arts");

     printBook(&book1);
     return 0;
}
```

```
void printBook(struct Book *book) {
     printf(“Book title: %s\n”, book->title);
     printf(“Book author: %s\n”, book->author);
     printf(“Book subject: %s\n”, book->subject);
     printf(“Book identifier: %d\n”, book->book_id);
}
```

Putem adăuga la structură pointeri la funcţii

Exemplu:

```
struct Book {
     int book_id;
     char title[50];
     char author[50];
     char subject[100];

void (*printBook)(struct Book *book);
};


void printBook1(struct Book *book) {
     printf("Book title: %s\n", book->title);
     printf("Book author: %s\n", book->author);
     printf("Book subject: %s\n", book->subject);
     printf("Book identifier: %d\n", book->book_id);
}

void printBook2(struct Book *book) {
     printf("Details: %s, %s, %s,   %d    ", book->title, book-
>author, book->subject, book->book_id);
}

int main () {

/* declaratii, initalizari,… citiri */

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

switch(x)
{
    case 1:
    book1.printBook=&printBook1;
    break;
    default:
    book1.printBook=&printBook2;
}
```

```
book1.printBook(&book1);

return 0;
}
```

Accesarea membrilor structurilor prin intermediul pointerilor utilizand alocarea dinamica:

Putem aloca memorie dinamic utilizand functia malloc()din header-ul “stdlib.h”
Sintaxa:

```
ptr = (cast_type*)malloc(byte_size);
```

Exemplu:

```
int main() {
     struct Book *book_ptr;
     int i, n;

     printf(“Enter number of books: “);
     scanf(“%d, &n);

     //alocam memorie pentru n structuri Book cu book_ptr
     //indicand spre adresa de baza
     book_ptr = (struct Book*)malloc(n * sizeof(struct Book));

     for(i = 0; i < n; ++i) {
          printf(“Enter book id, title, author and subject“);
          scanf(“%d”, &(book_ptr+i)->book_id);
          scanf(“%s”, &(book_ptr+i)->title);
          scanf(“%s”, &(book_ptr+i)->author);
         scanf(“%s”, &(book_ptr+i)->subject);
     }
     //afisam informatiile despre cartile citite anterior
     for(i = 0; i < n; ++i) {
          printBook1(book_ptr+i);
     }

     return 0;
}
```

Siruri de caractere

Limbajul C nu dispune de un tip de date nativ pentru reprezentarea sirurilor de caractere de lungime variabila. In acest scop se utilizeaza structuri de date de tip tablou de caractere.

Intrucat sirurile de caractere prelucrate in programe au in general lungime variabila, s-a stabilit o conventie prin care ultimul caracter utilizat al unui sir este urmat de un caracter cu valoarea zero ('\0'), numit terminator de sir.

```
char sir [10];
```

Exemplul de mai sus declara un tablou de 10 de elemente de tip caracter. Un asemenea tablou se poate folosi pentru memorarea unui sir de caractere de lungime variabila, dar de maxim 9 de caractere, intrucat ultimul element este rezervat pentru terminatorul de sir.

Daca sirul de mai sus contine valoarea "TEST", continutul memoriei rezervate tabloului este urmatorul:

Index	0	1	2	3	4	5	6	7	8	9
Continut	T	E	S	T	**\0**	-	-	-	-	-

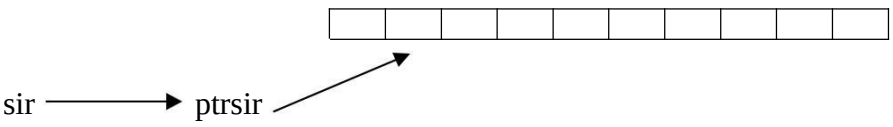
Valoarea elementelor tabloului corespunzatoare indicilor de la 5 la 9 nu este cunoscuta. Evident, aceste tablouri care memoreaza siruri de caractere se pot aloca si dinamic:

```
char *sir = (char*)malloc (10 * sizeof (char));
/* ...operatii cu variabila sir... */
free (sir);
```

Daca se doreste doar accesarea si prelucrarea elementelor unui sir de caractere care a fost alocat anterior, static (declaratie de tablou) sau dinamic, se poate utiliza si o variabila de tip pointer catre caracter:

```
char sir [10];
char *ptrsir;
ptrsir = sir;
```

/*Ambele variabile indica spre acelasi sir de caractere*/



Orice sir de caractere care se prelucreaza in program trebuie sa dispuna insa de o declaratie de alocare de memorie (static sau dinamic).

Compilatoarele de C permit initializarea tablourilor de caractere in momentul declararii acestora cu un sir de caractere:

```
char sir [10] = "Un sir";
```

In urma acestei declaratii, variabila *sir* va avea urmatorul continut:

Index	0	1	2	3	4	5	6	7	8	9
Continut	U	n		S	i	r	**\0**	-	-	-

Lista funcțiilor pentru siruri de caractere:
http://www.cplusplus.com/reference/cstring/

Probleme

1. Sa se construiasca o structura pentru a reprezenta numerele complexe.
   a) Adăugați în structură pointeri la funcții de afișare și citire ;
   b) Utilizand pointeri si functii definite in afara structurii, efectuati adunarea, inmultirea, conjugarea a n numere complexe.
   c) Scrieţi o funcție generică ce va avea ca parametric un număr complex și un parametru de tip pointer la funcţie. Folosind aceasta funcţia generică, tabelaţi cele trei operații matematice de la pct b).

2. Sa se cate o funcție C pentru fiecare din următoarele cerințe:
   a) Sterge dintr-un sir de caractere un subsir specificat prin pozitie si lungime si returnează noul șir;

      ```
      char *sterg(char *p, int poz, int cate);
      ```

   b) Inserează într-un șir începând cu o poziție dată un al șir.

      ```
      char *inserez(char *p, char *s);
      ```

   c) Citeste doua cuvinte si inlocuieste intr-un text introdus de la tastatura toate aparitiile primului cuvant prin cel de-al doilea.
   d) Folosind funcția `strtok` citeste o fraza si afiseaza pe cate un rand cuvintele sale (fara semne de punctuatie sau spatii).

Tema 2:

https://docs.google.com/document/d/1VMhaW7_5eQS-BQToqUyNc6JDhglwya_kZHZnrGuQeiI/edit