Structuri în limbajul de programare C

Structurile sunt tipuri de date în care putem grupa mai multe variabile eventual de tipuri diferite (spre deosebire de vectori, care conţin numai date de acelaşi tip). Numele structurii nu este un nume de variabilă, ci un nume de tip de date care se mai numeşte şi eticheta structurii. Instrucţiunile de definire a structurilor se vor plasa la începutul fişierului în afara oricărei funcţii.

O structură se poate defini astfel:

```
struct nume_structura {
  declaratii_de_variabile
};
```

Exemplu de ***declarare de structură***:

```
struct student {
  char numePrenume[10];
  int varsta;
  float notaMedie;
};
```

Variabilele declarate în interiorul structurii se numesc "câmpurile" structurii: ***numePrenume[10], varsta*** și ***notaMedie***, iar ***student*** este eticheta structurii, adică numele structurii și a tipului de date definit și stabilit de programator.

Exemplu de program cu tip de date structură:

```
#include <stdio.h>
#include <string.h>

struct student {
  char numePrenume[25];
  int varsta;
  float notaMedie;
};

int main()
{
    struct student s;
    // "%[^\n]%*c"  se va citi caracterele introduse de la tastatura
    //  pana cand nu se va apasa tasta Enter
    printf("Numele: ");
    scanf ("%[^\n]%*c", s.numePrenume);
    // "%s" se va citi caracterele pana la primul caracter alb (spatiu sau Enter)
    //scanf ("%s", s.numePrenume);
    printf("Varsta: ");
    scanf("%d", &s.varsta);
    printf("Nota Medie: ");
    scanf("%f", &s.notaMedie);

    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.", s.numePrenume, s.varsta, s.notaMedie);
    return 0;
}
```

O altă alternativă ar fi:

```
#include <stdio.h>
#include <string.h>

typedef struct student {
  char numePrenume[25];
  int varsta;
  double notaMedie;
} Stud;

int main()
{
    Stud s;

    printf("Numele: ");
    scanf ("%[^\n]%*c", s.numePrenume);

    printf("Varsta: ");
    scanf("%d", &s.varsta);
    printf("Nota Medie: ");
    scanf("%lf", &s.notaMedie);

    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2lf.", s.numePrenume, s.varsta, s.notaMedie);

    return 0;
}
```

Declararea variabilelor de tip structură

Forma generală:

```
struct nume_structura {
 declaratii_de_variabile
} [one or more structure variables];
```

Exemplu de declarare de structură:

```
struct student {
  char numePrenume[10];
  int varsta;
  float notaMedie;
} s1, s2;
```

Exemplu de program cu tip de date structură:

```
#include <stdio.h>
#include <string.h>

struct student {
  char numePrenume[25];
  int varsta;
  float notaMedie;
} s1;
```

```
int main()
{
    printf("Numele: ");
    scanf ("%[^\n]%*c", s1.numePrenume);

    printf("Varsta: ");
    scanf("%d", &s1.varsta);
    printf("Nota Medie: ");
    scanf("%f", &s1.notaMedie);

    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.", s1.numePrenume, s1.varsta, s1.notaMedie);
    return 0;
}
```

Inițializarea unor variabile de tip structură:

```
struct student s1 = {"Carp Ion", 30, 8.85};
```

sau:

```
struct student s1 = {.numePrenume = "Ursu Ana", .varsta = 25, .notaMedie = 9.7568};
```

sau, inițializarea fiecărui câmp separat:

```
struct student s1 = {.numePrenume = "Ursu Ana", .varsta = 25, .notaMedie = 9.7568};
```

Exemplu de program:

```
#include <stdio.h>
#include <string.h>

struct student {
  char numePrenume[25];
  int varsta;
  float notaMedie;
};

int main()
{
    struct student s = {"Carp Ion", 30, 8.85};
    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.\n", s.numePrenume, s.varsta, s.notaMedie);

    struct student s1 = {.numePrenume = "Ursu Ana", .varsta = 25, .notaMedie = 9.7568};
    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.", s1.numePrenume, s1.varsta, s1.notaMedie);

    return 0;
}
```

Declararea structurilor pot fi făcute și într-un ***fișier antet creat de programator***.
Exemplu de program:

```
//continutul fisierului creat de programator
```

```
#ifndef st  //denumirea fisierului este st.h
#define st

struct student {
  char numePrenume[25];
  int varsta;
  float notaMedie;
};

#endif
```

Programul principal:

```
#include <stdio.h>
#include <string.h>
#include "st.h"  //fisierul creat de utilizator

int main()
{
    struct student s = {"Carp Ion", 30, 8.85};
    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.\n", s.numePrenume, s.varsta, s.notaMedie);

    struct student s1 = {.numePrenume = "Ursu Ana", .varsta = 25, .notaMedie = 9.7568};
    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.", s1.numePrenume, s1.varsta, s1.notaMedie);

    return 0;
}
```

Inițializarea datelor unei structuri din funcții folosind **pointerii** și **returnarea unei structuri din funcție**.

```
#ifndef st   //denumirea fisierului este st.h
#define st

typedef struct student {
  char numePrenume[25];
  int varsta;
  float notaMedie;
} Student;

#endif
```

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "st.h"

void readData(Student *s);
Student getData();

int main()
```

```
{
    //initializare date apeland functie readData() si folosind pointeri
    Student *s = (Student *) malloc (sizeof(Student));
    readData(s);
    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.\n", s->numePrenume, s->varsta, s->notaMedie);
    free(s);
    //initializare date prin returnarea unei structuri din functia getData();
    Student s1 = getData();
    printf("\n%s:\nVarsta: %d;\nNota Medie: %.2f.", s1.numePrenume, s1.varsta, s1.notaMedie);

    return 0;
}

void readData(Student *s){
    strcpy(s->numePrenume, "Carp Ion");
    s->varsta =  30;
    s->notaMedie = 8.85;
}

Student getData(){
    Student s;
    strcpy(s.numePrenume, "Ursu Ana");
    s.varsta =  25;
    s.notaMedie = 9.7568;

    return s;
}
```

Citirea datelor unei **structuri de la tastatura** într-o funcție creată de utilizator, folosind **pointerii**. Crearea și utilizarea **fișierelor antet definite de utilizator**. **Sortarea Bubble** după câmpul **numePrenume**.

```
#ifndef st   //denumirea fisierului antet este st.h
#define st

void removeChar(char * str, char charToRemmove){
    int i, j;
    int len = strlen(str);
    for(i=0; i<len; i++)
    {
        if(str[i] == charToRemmove)
        {
            for(j=i; j<len; j++)
            {
                str[j] = str[j+1];
            }
            len--;
            i--;
        }
    }
```

```
}

void clearBuffer() {
        char c;
        do {
                c = getchar();
        } while (c != '\n' && c != EOF);
}
#endif
```

Programul principal:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>
#include "st.h"
//#define n 2

typedef struct student {
  char numePrenume[25];
  float notaMedie;
} Student;

void readData(Student *st, int n);
void writeData(Student *st, int n);
void bubbleSort(Student* std, int n);
void swap(Student *x, Student *y);

int main()
{
    Student *std = NULL;
    int i, n;
    printf("n="); scanf("%i", &n);
    clearBuffer();
    std = (Student *) calloc(n, sizeof(Student));

    readData(std, n);
    printf("\n---------afisare date-----------\n");
    writeData(std, n);
    bubbleSort(std, n);
    printf("---------afisare date dupa sortare-----------\n");
    writeData(std, n);
    free(std);
  return 0;
}
void readData(Student *st1, int n){

    for(int i = 0; i < n; ++i){
        printf("Nume & Prenume: ");
        scanf("%[^\n]%*c", (st1 + i)->numePrenume);
```

```
//      fgets(st1[i].numePrenume, sizeof(st1[i].numePrenume), stdin); //fflush(stdin);
//      fgets(st1[i].numePrenume, 25, stdin);
//      removeChar(st1[i].numePrenume, '\n');
        printf("Nota medie: "); scanf("%f", &(st1 + i)->notaMedie);
        clearBuffer();
    }
}
void writeData(Student *st1, int n){
    for(int i = 0; i < n; ++i){
        printf("%s: %g\n", (st1 + i)->numePrenume, (st1 + i)->notaMedie);
    //puts((st1 + i)->numePrenume);
    }
}
void bubbleSort(Student* std, int n){
    bool flag=true;
    Student temp;
    printf("\n\nLista studentilor in ordinea alfabetica a numelui\n");
    while(flag){
        flag=false;
        for(int i=0;i<n-1;i++){
            if(strcmp(std[i].numePrenume, std[i+1].numePrenume)>0){
                temp = std[i];    // temp = *(std + i);
                std[i]=std[i+1];  // *(std + i) = *(std + i + 1);
                std[i+1]=temp;    // *(std + i + 1) = temp;
                flag=true;
            }
        }
    }
}
void swap(Student *x, Student *y){
    Student temp;
    temp = *x;
    *x = *y;
    *y = temp;
}
void bubbleSort(Student* std, int n){
    bool flag=true;
    printf("\n\nLista studentilor in ordinea alfabetica a numelui\n");
    while(flag){
        flag=false;
        for(int i=0;i<n-1;i++){
            if(strcmp(std[i].numePrenume, std[i+1].numePrenume)>0){
               // swap(&std[i], &std[i+1]);
               swap((std+i), (std+i+1));
                flag=true;
            }
        }
    }
}
```

Manipularea datelor unui **tabloul unidimensional** cu elemente de tip **structură fără pointeri**:

```
#include <stdio.h>
#include <string.h>

struct student{
    char nume[20];
    float nota;
};

void read(int n);
void write(int n);
void bubbleInt(int n);
void bubbleChar(int n);
void heapSort(int n);
void clearBuffer();

struct student st[10];
struct student aux;

int main()
{
    int n;
    printf("n=");
    scanf("%d", &n);
    clearBuffer();
    read(n);
    printf("--------afisare date nesortate-----------\n");
    write(n);
    bubbleInt(n);
    printf("--------afisare date dupa sortare bubbleInt-----------\n");
    write(n);
    bubbleChar(n);
    printf("--------afisare date dupa sortare bubbleChar-----------\n");
    write(n);
    heapSort(n);
    printf("--------afisare date dupa sortare heapSort-----------\n");
    write(n);

    return 0;
}
void read(int n){
    for(int i = 0; i < n; ++i){
        printf("Nume: "); scanf("%[^\n]%*c", st[i].nume);
        printf("Nota: "); scanf("%f", &st[i].nota);
        clearBuffer();  //fflush(stdin);
    }
}

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

```
        printf("%s: ", st[i].nume);
        printf("%.2f\n", st[i].nota);
    }
}

void clearBuffer(){
    char c;
    do {
        c = getchar();
    } while(c != '\n');
}

void bubbleInt(int n){
    int i,schimbat;
    do
    {
        schimbat = 0;
        for(i = 0; i < n-1; i++)
            if(st[i].nota > st[i+1].nota) {
                aux = st[i];
                st[i] = st[i+1];
                st[i+1] = aux;
                schimbat = 1;
            }
    }while(schimbat);
}
void bubbleChar(int n){
    int i,schimbat;
    do
    {
        schimbat = 0;
        for(i = 0; i < n-1; i++)
            if(strcmp(st[i].nume, st[i+1].nume)>0) {
                aux = st[i];
                st[i] = st[i+1];
                st[i+1] = aux;
                schimbat = 1;
            }
    }while(schimbat);
}
void heapify(int n, int i) {
    int largest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;

    if (left < n && st[left].nota > st[largest].nota)
      largest = left;

    if (right < n && st[right].nota > st[largest].nota)
      largest = right;
```

```
    if (largest != i) {
        aux = st[i];
        st[i] = st[largest];
        st[largest] = aux;

      heapify(n, largest);
    }
}
void heapSort(int n) {
    for (int i = n / 2 - 1; i >= 0; i--)
      heapify(n, i);

    for (int i = n - 1; i >= 0; i--) {
        aux = st[0];
        st[0] = st[i];
        st[i] = aux;
      heapify(i, 0);
    }
  }
```