Структуры в Си

Лекция 10

Структуры

Структура — это объединение нескольких объектов, возможно, различного типа под одним именем, которое является типом структуры. В качестве объектов могут выступать переменные, массивы, указатели и другие структуры.

Определение из Wiki:

В языке Си, структура (struct) — композитный тип данных, инкапсулирующий без сокрытия набор значений различных типов. Порядок размещения значений в памяти задаётся при определении типа и сохраняется на протяжении времени жизни объектов, что даёт возможность косвенного доступа (например, через указатели)

Синтаксис объявления

Общая форма объявления структуры:

```
struct тип_структуры
{
  тип ИмяЭлемента1;
  тип ИмяЭлемента2;
  . . .
  тип ИмяЭлементаn;
};
```

После закрывающей фигурной скобки } в объявлении структуры обязательно ставится точка с запятой.

Пример

```
struct point_t {
    int x;
    int y;
};
```

```
struct addr {
        char name[30];
        char street [40];
        char city[20];
        char state[3];
        unsigned int zip;
};
```

```
struct date
{
  int day;
  char *month;
  int year;
};
```

Обычно все члены структуры связаны друг с другом.

Переменные типа «структура»

После того, как мы объявили структуру, можно создавать переменную такого типа с использованием служебного слова struct. Для объявления настоящей переменной, соответствующей структуре, следует написать:

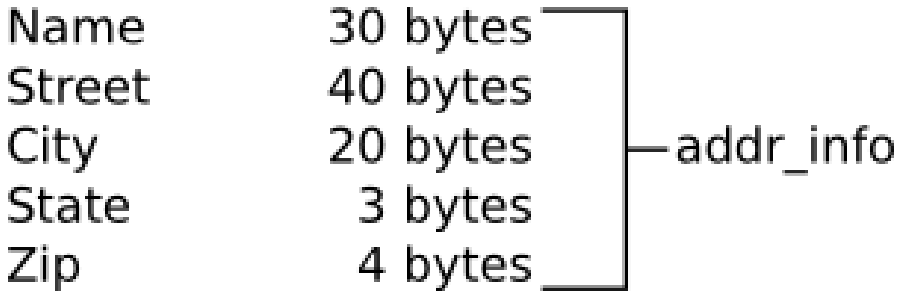
```
struct addr addr_info;
```

При объявлении структуры можно одновременно объявить одну или несколько переменных.

```
struct addr {
        char name[30];
        char street[40];
        char city[20];
        char state[3];
        unsigned int zip;
} binfo, cinfo;
```

Размещение в памяти

```
struct addr {
        char name[30];
        char street [40];
        char city[20];
        char state[3];
        unsigned int zip;
};
```



Доступ к полям структуры

Доступ до полей структуры осуществляется с помощью операции точка

```c
#include <stdio.h>
#include <math.h>

struct point_t {
    int x;
    int y;
};
void main() {
    struct point_t A;
    float distance;

    A.x = 10;
    A.y = 20;

    distance = sqrt((float) (A.x*A.x + A.y*A.y));

    printf("x = %.3f", distance);
    getchar();
}
```

```c
#include <stdio.h>
#include <math.h>

void main() {
    struct point_t {
        int x;
        int y;
    };
    struct point_t A;
    float distance;

    A.x = 10;
    A.y = 20;

    distance = sqrt((float) (A.x*A.x + A.y*A.y));

    printf("x = %.3f", distance);
    getchar();
}
```

```c
#include <stdio.h>
#include <math.h>

void main() {
    struct point_t {
        int x;
        int y;
    } A;
    float distance;

    A.x = 10;
    A.y = 20;

    distance = sqrt((float) (A.x*A.x + A.y*A.y));

    printf("x = %.3f", distance);
    getchar();
}
```

Анонимная структура

Структура может быть анонимной. Тогда ее не возможно использовать в дальнейшем.

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

void main() {
    struct {
        int x;
        int y;
    } A;
    float distance;

    A.x = 10;
    A.y = 20;

    distance = sqrt((float) (A.x*A.x + A.y*A.y));

    printf("x = %.3f", distance);
    getchar();
}
```

Инициализация полей структуры

Инициализация полей структуры может осуществляться двумя способами:

- присвоение значений элементам структуры в процессе объявления переменной, относящейся к типу структуры;
- присвоение начальных значений элементам структуры с использованием функций ввода-вывода (например, printf() и scanf()).

Инициализация полей структуры

```c
#include <stdio.h>
#include <math.h>

struct gasket {
    float weight;
    unsigned height;
    unsigned diameter;
};

void main() {
    struct gasket obj = { 12.f, 120, 30 };

    printf("gasket info:\n");
    printf("------------------\n");
    printf("weight: %4.3f kg\n", obj.weight);
    printf("height: %6d cm\n", obj.height);
    printf("diameter: %4d cm\n", obj.diameter);

    getchar();
}
```

```c
#include<stdio.h>

typedef struct thing {
    int a;
    float b;
    const char *c;
} thing_t;

int main() {
    thing_t t = {
        .a = 10,
        .b = 1.0,
        .c = "olololololo"
    };
    printf("%s\n", t.c);
    printf("%d\n", t.a);
    printf("%f\n", t.b);
    getchar();
}
```

Инициализация полей структуры

```c
#include <stdio.h>
#include <stdlib.h>

struct date {
  int day;
  char month[20];
  int year;
};
struct persone {
  char firstname[20];
  char lastname[20];
  struct date bd;
};
```

```c
void main() {
  struct persone p;
  printf("Enter name : ");
  scanf("%s", p.firstname);
  printf("Enter surname : ");
  scanf("%s", p.lastname);
  printf("Enter birthday\nDay: ");
  scanf("%d", &p.bd.day);
  printf("Month: ");
  scanf("%s", p.bd.month);
  printf("Year: ");
  scanf("%d", &p.bd.year);
  printf("\nYou enter : %s %s, birthday %d %s %d",
    p.firstname, p.lastname, p.bd.day, p.bd.month, p.bd.year);
  getchar(); getchar();
}
```

Определение нового типа

При определении новой структуры с помощью служебного слова struct, в пространстве имён структур создаётся новый идентификатор. Для доступа к нему необходимо использовать служебное слово struct.

Можно определить новый тип с помощью служебного слова typedef. Тогда будет создан псевдоним для структуры, видимый в глобальном контексте.

```
struct point_t {
    int x;
    int y;
};

typedef struct point_t Point;
```

Определение нового типа

```
#include <stdio.h>

struct point_t {
    int x;
    int y;
};

typedef struct point_t Point;

void main() {

    struct point_t p = {10, 20};

    Point px = {10, 20};

    getchar();
}
```

Определение нового типа

Теперь при работе с типом Point нет необходимости каждый раз писать слово struct. Два объявления можно объединить в одно

```
typedef struct point_t {
    int x;
    int y;
} Point;
```