Информатика

Кочанов Марк

17 ноября 2018 г.

МИФИ

Указатели (опять)

Пример

```c
#include <stdio.h>      /* printf */
#include <stdlib.h>     /* qsort */

// void qsort (void* base, size_t num, size_t size, int
↪  (*compar)(const void*,const void*));

int values[] = { 40, 10, 100, 90, 20, 25 };

int compare (const void * a, const void * b)
{
  return ( *(int*)a - *(int*)b );
}

int main ()
{
  int n;
  qsort (values, 6, sizeof(int), compare);
  for (n=0; n<6; n++)
     printf ("%d ",values[n]);
  return 0;
}
```

Указатели на функции

В языке программирования C имеется возможность создавать указатели не только на типы данных, но и на функции.

```c
void fun(int a)
{
    printf("Value of a is %d\n", a);
}

int main(void)
{
    void (*fun_ptr)(int) = fun;
    fun_ptr(10);

    return 0;
}
```

Как на самом деле

При инициализации указателя на функцию имя функции неявным образом приводится к указателю на эту функцию, поэтому оператор взятия адреса и оператор разыменования опциональны[1].

```
1  void fun(int a) {}
2
3  int main()
4  {
5      void (*fun_ptr)(int) = fun;
6      fun_ptr(10);
7
8      void (*fun_ptr_2)(int) = &fun;
9      fun_ptr_2(10);
10
11     void (*fun_ptr_3)(int) = ******fun;
12     (*fun_ptr_3)(10);
13
14     return 0;
15 }
```

[1] https://stackoverflow.com/q/6893285

Массив указателей на функции

```c
void add(int a, int b) { printf("a + b = %d\n", a + b); }
void subtract(int a, int b) { printf("a - b = %d\n", a - b); }
void multiply(int a, int b) { printf("a * b = %d\n", a * b); }

int main()
{
    void (*fun_ptr_arr[])(int, int) = {add, subtract,
    ↪  multiply};
    int a = 15, b = 10;

    fun_ptr_arr[0](a, b);
    fun_ptr_arr[1](a, b);
    fun_ptr_arr[2](a, b);

    return 0;
}
```

Передача функции в качестве аргумента

```c
void fun1() { printf("Fun1\n"); }
void fun2() { printf("Fun2\n"); }

// A function that receives a function
// as parameter and calls the function
void wrapper(void (*fun)())
{
    fun();
}

int main()
{
    wrapper(fun1);
    wrapper(fun2);

    return 0;
}
```

Указатели посложнее

Например, можно строить указатели на функции, аргумент которой есть массив указателей на функции, каждая из которых возвращает указатели на другие функции[2].

```
1  // declare func_1 as pointer to function (void) returning
   ↪  pointer to array 3 of int
2  int (*(*func_1)(void ))[3]
3
4  // declare func_2 as function (int, pointer to function (int)
   ↪  returning void) returning pointer to function (int)
   ↪  returning void
5  void (*func_2(int, void (*)(int)))(int);
6
7  // declare func_3 as array 3 of pointer to function returning
   ↪  pointer to array 5 of char
8  char (*(*func_3[3])())[5]
```

[2] https://cdecl.org

Задача

Функции для обработки массивов

Реализовать функции map, filter, reduce, встречаемые во многих языках программирования и предднаначенные для обработки массивов.

Функция map() должна принимать на вход массив чисел типа int и указатель на функцию, принимающую на вход int и возвращающую int. В процессе работы функция map() применяет к каждому элементу массива переданную функцию и возвращает в качестве результата массив полученных чисел.

Например, в функцию map() был передан массив чисел [2, 5, 3] и функция возведения в квадрат. Результатом работы функции map() будет **новый** массив, содержащий числа [4, 25, 9].

Задача (продолжение)

Функция filter() должна принимать на вход массив чисел типа int и указатель на функцию, принимающую на вход int и возвращающую int (0 или 1). В процессе работы функция filter() проходится по массиву переданных чисел и добавляет число в результирующий массив если результат применения функции к числу вернул 1.

Например, в функцию filter() был передан массив чисел [2, 5, 3, 6] и функция, возвращающая 1, если число четное. Результатом работы функции filter() будет **новый** массив, содержащий числа [2, 6].

Замечание: функция filter() может возвращать в качестве результата длину полученного массива, а указатель на сам массив возвращать через переданный указатель.

Задача (продолжение)

Функция reduce() должна принимать на вход массив чисел типа int, начальное значения «аккумулятора» и указатель на функцию, принимающую на вход два числа типа int и возвращающую int. В процессе работы функция reduce() проходится по массиву переданных чисел, вызывает для каждого элемента переданную функцию, передавая в качестве первого рагумента текущее значения аккумулятора, а в качестые второго аргумента — текущее число из массива. Результат работы функции становится новым значения аккумулятора. В конце работы функия reduce() возвращает значение аккумулятора.

Например, в функцию reduce() был передан массив чисел [2, 5, 3], число 0 в качестве начального значения аккумулятора и функция сложения двух чисел. Результатом работы функции reduce() сумма элементов массива. Пошаговый алгоритм работы функции:
аккумулятор=0 число=2 аккумулятор после вызова функции=2
аккумулятор=2 число=5 аккумулятор после вызова функции=7
аккумулятор=7 число=3 аккумулятор после вызова функции=10
результат работы функции reduce()=10