|  |  |
| --- | --- |
| Здравствуйте, коллеги, рад всех приветствовать на восьмом занятии по Основам языка С. На этом уроке нас с вами ждут массивы и ещё пара слов о директивах компилятору, также называемых директивами препроцессора. С них и начнём. |  |
| Помимо уже хорошо знакомой вам директивы #include естественно, существуют и другие. Некоторые вы могли заметить в созданном для нас средой разработки заголовочном файле. Они ограничивают импорт описанных в заголовочном файле функций, дословно (если ещё не описаны, описать, конец условия). | #ifndef  #define  #endif |
| Вот, директиву ОПИСАТЬ мы и рассмотрим подробнее. Она не зря называется директивой препроцессора, поскольку даёт указание не процессору во время выполнения программы выделить память, присвоить значения, а непосредственно компилятору - заменить в тексте программы одни слова на другие. Например, таким образом можно задавать константы проекта, напишем #define ARRAY\_LENGTH 50 и это будет означать, что компилятор, перед тем как запустить нашу программу заменит все слова ARRAY\_LENGTH на цифру 50. Весьма удобно, но этим можно не ограничиваться, мы можем попросить компилятор заменить целые вызовы функций и операторы на короткие, удобные нам слова. Иногда в программах можно встретить описание недостающего но такого привычного булевого типа при помощи директив препроцессора #define boolean int #define true 1 #define false 0. Но нам с вами пока что достаточно умения создать глобальную именованную константу. | #define ARRAY\_LENGTH 50  int a = ARRAY\_LENGTH;  #define boolean int  #define true 1  #define false 0 |
| Перейдем к обещанным массивам. Массив – это множество данных одного типа. Язык С строго типизирован, поэтому невозможно создать массив из разных типов данных. На данном этапе мы рассматриваем только статические типы данных, поэтому и массивы будем рассматривать статические. Статическим массивом называют массив, количество элементов которого заранее известно и не изменяется. Статические массивы принято объявлять двумя способами: |  |
| здесь мы объявляем массив, содержащий элементы типа int, имя массива arr, количество элементов массива 3.  Как уже говорилось массив это множество данных или элементов. К каждому элементу массива можно обратиться по его номеру, который принято называть индексом. Индексация элементов начинается с нуля. | int arr[3]; |
| Давайте заполним наш массив значениями типа int. Для этого последовательно обращаемся к каждому элементу и присваиваем значение: | arr[0] = 10;  arr[1] = 20;  arr[2] = 30; |
| Обратите внимание, что язык С не гарантирует что инициализационное значение элементов массива будет равно нулю, если это не указано явно. |  |
| Второй способ объявления и инициализации массива используют, если массив небольшой и его значения заранее известны: | int arr[5] = {0, 1, 2, 3, 4}; |
| Итак, мы научились создавать и заполнять значениями массивы. Теперь общее правило объявления массивов в Си: при объявлении массива нужно указать его имя, тип элементов, количество элементов. Количество элементов есть натуральное число, т.е. целое положительное. Ноль не может быть количеством элементов. Нельзя задавать переменное количество элементов массива. | int nArr[100]; // Объявлен массив, предназначенный для хранения ста целых чисел;  float fArr[5]; // Объявлен массив, предназначенный для хранения 5-ти чисел типа float;  char cArr[2]; // Объявлен массив, предназначенный для хранения двух символов;  int varElem;  int nArr[varElem]; // Ошибка! Количество элементов нельзя задавать переменной; |
| Теперь давайте научимся получать доступ к элементам массива. Нет ничего проще, для этого нужно указать имя массива и индекс элемента: | int a = arr[0] - получить значение 0-го элемента массива arr и присвоить его переменной а. |
| При помощи массивов решают множество задач, таких как поиск, сортировка, составление таблиц соответствия, создание частотных диаграмм. Для примера давайте напишем программу, которая будет печатать наш массив в консоль. | #include <stdio.h>  int main( int argc, char \*argv[])  {  int i;  printf("Your array is: ");  for (i = 0; i < 5; i++)  {  printf("%d ", arr[i]); // получаем значение i элемента массива и выводим в консоль.  }  return 0;  } |
| Мы научились создавать, инициализировать массивы и обращаться к его элементам. Теперь решим задачу посложнее - напишем программу, которая проверит насколько статистически хорош описанный в стандартной библиотеке генератор случайных чисел. Для такой статистической проверки нам понадобится сформировать так называемый частотный массив. Массив, в котором будет содержаться информация о том, сколько раз то или иное число появилось во множестве, полученном при помощи генератора псевдослучайных чисел.  Данная программа наглядно демонстрирует не только работу с массивами, но и то, что генератор псевдослучайных чисел в языке С генерирует статистически верную последовательность случайных чисел. | #include <stdio.h>  #include <stdlib.h>  #include <time.h>  #define ARRAY\_LENGTH 10  #define NUMBERS\_AMOUNT 1000000;  int main( int argc, char \*argv[])  {  srand(time(NULL)); // зададим начальное значение ГПСЧ  int frequency[ARRAY\_LENGTH] = {0}; // объявим и обнулим  int a, i;  for (i = 0; i < NUMBERS\_AMOUNT; i++) // заполним  {  a = rand() % ARRAY\_LENGTH; // сгенерируем  frequency[a]++; // для числа 0 увеличим значение в 0-й ячейке массива, для числа 1 - в 1-й, и т.д.  }  for (i = 0; i < ARRAY\_LENGTH; i++) // выведем  {  printf("Number %d generated %6d (%5.2f%%) times\n", i, frequency[i], ((float)frequency[i] / NUMBERS\_AMOUNT \* 100)); // количество вхождений и подсчёт процентов  }  return 0;  } |
| До встречи на следующем занятии. |  |