Семинар 4

Это задание будет в формате лабораторной работы и посвящено работе с указателями и ссылками. В качестве отчета пришлите файл с исходным кодом, в котором будут решения с пояснениями.

Задача 1. Создайте массив целых чисел. Выведите адрес начала массива.

Выведите адрес каждого элемента массива. Рассмотрите разности адресов соседних элементов. Чему они равны? Что они показывают? Почему при каждом запуске разные числа? Как разности соотносятся с sizeof ? Что показывает разница между двумя указателями? Приведите указатели к типу long int и еще раз посмотрите разницу между указателями на соседние элементы массива. Что она показывает?

Проведите такой же эксперимент для чисел типа double. Что изменилось и почему?

Задача 2. Заведите переменную x целого типа и инициализируйте ее каким-нибудь числом. После чего заведите указатель px на эту переменную.

Выведите следующие значения: x, &x, px, &px, *px, &*px.
Почему значения совпали/не совпали? Объясните полученные числа.

Задача 3. Напишите функцию, со следующим прототипом:

void PtrTest(int number, int* pointer)
Функция должна уметь выводить адреса и значения переменных. Для pointer помимо этого должно выводиться значение той ячейки, на которую он указывает.

Убедитесь в том, что при вызове функции происходит копирование аргументов, а именно:

1) для переменных выделяется новая память (обратите внимание на адреса в памяти);

2) значения переменных вызывающей функции (main), задействованных при вызове функции PtrTest, совпадает со значениями аргументов функции.

Измените теперь функцию PtrTest так, чтобы она возвращала адрес переменной number. Скомпилируйте код. Что вывел компилятор? В чем заключается опасность возвращения адреса локальной переменной?

Задача 4. Создайте массив целых чисел m со значениями [10, 20, 30, 40].

Объясните значения следующих выражений

- *m,
- &m[0],
- *(m+1),
- m,
- &*m,
- &*&*m,
- (m + 1)[1],

- *(m + 2), *(2 + m),
- m[2], 2[m]

Задача 5. Создайте строку длиной 200 млн. символов (например, так: std::string str(200000000, ‘d’). Не забудьте сделать #include <string>). Напишите функцию, которая принимает на строку по значению и изменяет первый символ строки на какой-то другой. Используя созданную строку, вызовите функцию 100 раз в цикле.
Засеките время её выполнения (для этого можно использовать код отсюда https://github.com/khalkechev/cpp/blob/master/hse/2016_spring/seminars/seminar_03/src/seminar-03-task-02-experiment.cpp или на linux-подобных машинах можно из консоли использовать команду time, например, так: time ./a.out, где a.out - исполняемый файл).
Если программа работает слишком долго (больше минуты), уменьшите размер строки, если слишком быстро (доли секунды) - увеличьте.

Теперь передайте строку по ссылке. Изменилось ли время исполнения?
А если заменить на указатель? Сравните время выполнения для трех запусков (передача по значению, указателю и ссылке), объясните полученные цифры.