|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| Предмет: ВП |  |
| Подготовил студент гр. ИУК4-12Б: | Бобров К.А. (https://t.me/I\_Am\_L0se) |

**Тема №11: Указатели и ссылки. Примеры.**

**Указатель — это обычная переменная, которая хранит адрес какого - то объекта. Указатель, как и любая другая переменная имеет свой адрес в памяти.**

// создание нулевого указателя

int \*ptr\_1{};

int \*ptr\_2 = nullptr;

**Для создания нескольких указателей, перед каждым именем нужно ставить оператор звëздочки.**

// создание нескольких указателей

int \*ptr\_1{}, \*ptr\_2{}, \*ptr\_3{};

// содание одного указателя

int \*ptr\_4{}, var\_1{}, var\_2{}

**Отличие const int \* от int \* const**

|  |  |
| --- | --- |
| const int \*ptr\_1 = &my\_var\_1; | int \* const ptr\_1 = &my\_var\_1; |
| Чтобы создать указатель, хранящий адрес константы, нужно чтобы этот указатель имел тип указателя на константу | Если мы не хотим менять значение указателя в течении выполнения программы, нужно создать константный указатель |

**Ссылка - это некий псевдоним для переменной (второе имя), не является типом переменной. Под капотом является константным указателем.**

// создание ссылки

int &link\_1 = my\_var\_1;

\* Данные взяты из «Лекция 11» Пчелинцевой Н. И.

**Передача по указателю и передача по ссылке**

В следующей таблице перечислены основные различия между методами передачи по указателю и передачи по ссылке.

| **Параметры** | **Передача по указателю** | **Передача по ссылке** |
| --- | --- | --- |
| **Передача аргументов** | Мы передаем адрес аргументов в вызове функции. | Мы передаем аргументы в вызове функции. |
| **Доступ к значениям** | Доступ к значению аргументов осуществляется через оператор разыменования **\*** | Ссылочное имя можно использовать для неявной ссылки на значение. |
| **Переназначения** | Переданные параметры могут быть перемещены/переназначены в другое место в памяти. | Параметры не могут быть перемещены/переназначены на другой адрес памяти. |
| **Допустимые значения** | Указатели могут содержать значение NULL, поэтому переданный аргумент может указывать на значение NULL или даже на мусорное значение. | Ссылки не могут содержать значение NULL, поэтому оно гарантированно имеет какое-то значение. |

**Разница между ссылочной переменной и переменной-указателем**

Ссылка — это тот же объект, только с другим именем, и ссылка должна ссылаться на объект. Поскольку ссылки не могут иметь значение NULL, они более безопасны в использовании.

* Указатель может быть переназначен, а ссылка — нет, и должен быть назначен только при инициализации.
* Указателю может быть присвоен NULL напрямую, в то время как ссылке — нет.
* Указатели могут перебирать массив, мы можем использовать операторы инкремента/декремента для перехода к следующему/предыдущему элементу, на который указывает указатель.
* Указатель — это переменная, содержащая адрес памяти. Ссылка имеет тот же адрес памяти, что и элемент, на который она ссылается.
* Указатель на класс/структуру использует '->' (оператор стрелки) для доступа к своим членам, в то время как ссылка использует '.' (оператор точки)
* Указатель должен быть разыменован с помощью \*, чтобы получить доступ к области памяти, на которую он указывает, в то время как ссылка может быть использована напрямую.

\* Данные взяты с сайта [Передача по указателю и передача по ссылке в C++ - GeeksforGeeks](https://www.geeksforgeeks.org/passing-by-pointer-vs-passing-by-reference-in-cpp/?ref=lbp)