Семинар 3.13

Указатели и ссылки

Ссылки

Ссылка - это "псевдоним" переменной. Ссылка обязательно инициализируется при объявлении.
Объявление:
тип &имя_ссылки = имя_переменной;

- Ссылка не может быть нулевой
- Ссылку нельзя переопределить (она всегда ссылается на одно и тоже место)

```
int main() {
    int b = 7, a = 5;
    int &p = a;
    cout << a << ' ' << p << endl; //выведет 5 5
    a = 6;
    cout << a << ' ' << p << endl; //выведет 6 6
    p = b;
    cout << a << ' ' << p << endl; //выведет 7 7
    cout << &a << ' ' << &p << endl; //?
}
```

Ссылка на функцию

Аналогично переменным, можно объявить ссылку на функцию.

```
void f (int x) {
    cout << x << endl;
}

int main(){
  void (&rf)(int) = f;//ссылка на функцию
   f(123); // выведет 123
   rf(321); //выведет 321
   return 0;
}
```

Указатели

Указатель - это самостоятельный тип данных. Указатель может указывать на разные сущности или быть нулевым (nullptr).

Объявление: **тип* имя_указателя;**

```
main() {
    int b = 7, a = 5; // объявляем переменные
    int* p ; // в p «мусор»!
    p = &a; // в p лежит адрес a
    cout << a << ' ' << *p << endl;
    a = 6;
    cout << a << ' ' << *p << endl;
    p = &b;
    cout << a << ' ' << *p << endl;
    cout << &a << ' ' << p << endl;
}
```

Динамическое выделение памяти

```
int n = 100500;
int* array = new int[n];
std::cout << " array =  " << array <<
"\n"; // Адрес array - из "кучи"

...
...

delete[] array; // Если мы выделяли
память через new, нужно освобождать её
вручную!
```