C++primer学习笔记

第二章 变量与基本类型

基本类型：字符、整型、浮点型等

整型：整数、字符、布尔统称为整型

有符号与无符号：signed(默认)与unsigned

越界处理：1.unsigned上越界：对取值个数取模

2.unsigned下越界：对取值个数取模

3.signed越界：取决于编译器

字符：char：8 bit　　wchar_t：16 bit

整型值（取决于机器字长）：

short int:半个机器字长

int：一个机器字长

longlong int：两个机器字长

浮点型：

float：保证6位有效数字

double：保证10位有效数字

字面值常量：

整型字面值：1.0开头表示8进制；

2.0x(0X)开头表示16进制；

3.后面加u(U)定义unsigned；

浮点字面值：1.科学计数法使用E(e);

2.单精度使用（F与f）；

3.双精度使用（L与l）；

布尔字面值：true与false

字符字面值：用一对单引号来定义

转义字符：反斜线加符号

字符串字面值：双引号括起来的一个或多个字符

多行字面值：一行末尾加反斜线；

变量：

左值：赋值语句的左边或右边；

右值：赋值语句的右边

变量名：变量的标识符，由数字（不能作为开头）、下划线、字母组成，区分大小写

注：数字不能用作标识符开头，关键字和操作符替代名不能作为标识符

变量初始化：1.=：复制初始值；

2.()：直接初始化；

初始化规则

内置类型的初始化：函数体外初始为0，函数体内不能自动初始化

类类型初始化：构造函数

变量的定义与声明：

变量的定义：为变量分配空间，只能定义一次

变量的声明：表明变量的类型和名字

作用域：全局作用域、局部作用域

const:常量关键字，不能修改

extern:定义整个程序变量关键字

引用：变量前加“&”，即对象的另一个名字

注：不能定义引用的引用，非$const$引用只能绑定到同类型对象

typedef：定义类型的同义词

```
typedef longlong LL
```

枚举：关键字enum,枚举成员是常量

类：以class关键字开始；

```
class MyClass {
    private:
    .........
    public:
    .........
};
```

第三章 标准库类型

string、vector、bitset等;

命名空间：使用using声明

访问std名称空间的4种方法：1.使用using namespace std 放在函数定义之前；

2.使用using namespace std 放在函数定义中；

3.特定的函数中使用类似 using std::cout的编译指令；

4.每次使用时加前缀std:: ;

getline:用于读取整行文本

string常用操作：1.s.empty()：判断空

2.s.size()：获取大小

3.s[n]：下标操作

4.s1+s2：字符串拼接

5.s1=s2：字符串复制

6.s1==s2：判断字符串是否相同

7.!=,<等：字符串比较

注：$size()$成员函数返回的实际上是一种配套类型：$size\_type$,这样库类型的使用就能与机器无关

vector初始化： 1.vector<int> vec;

2.vector<int> vec(v);

3.vector<int>(n,i);

4.vector<int>(n);

vector常用操作： 1.v.empty():判断是否为空

2.v.size()：返回大小

3.v.push_back()：添加元素

4.v[n]：下标操作

5.v1=v2：将v1中的元素替换为v2

6.v1==v2：判断是否相同

注：下标能否添加元素取决于容量大小

迭代器：iterator与const_ iterator（不能修改值）

```
for(vector<int>::iterator ite=vec.begin();ite=vec.end();ite++){.....};
```

bitset常用操作： 1.b.any():是否存在为1的二进制位

2.b.none()：是否不存在为1的二进制位

3.b.count():为1的二进制位个数

4.b.size()：二进制位个数

5.b[pos]：下标访问

6.b.test(pos)：pos处的二进制位是否为1

7.b.set()：所有二进制位置1

8.b.set(pos)：pos处的二进制位置1

9.b.reset()：所有二进制位置0

10.b.reset(pos)：pos处的二进制位置0

11.b.flip():所有二进制位逐位取反

12.b.flip(pos):pos处二进制位取反

第四章 数组与指针

数组： 类型名+标识符+维数

初始化方式：1.显示初始化：可以不指定维数

```
int array[]={1,2,3}
```

注：1.函数体外自动初始化为0，函数体内无自动初始化；2.数组不允许直接复制3.数组长度固定

指针： 用于指向对象，保存对象的地址。

有效指针的三种状态： 1.保存特定对象的地址；

2.指向特定对象后面的另一个对象；

3.0值(NULL)

void指针：可以保存任何类型对象的地址

注：*void*指针仅支持几种有限的操作：1.与另一个指针进行比较；2.传递与返回空指针；3.给另一个空指针赋值；4.不允许使用空指针操纵指向的

指向const的指针： 不能修改所指向对象的值；

```
const int n=5;
const int *p=&n;
```

const指针：不能修改所指向的对象；

```
int n=5;
int* const p=&n;
```

注：$int\ const*$与$const\ int*$意义一样

指针与引用的比较： 1.引用总是指向某个对象；

2.赋值行为的差异；

动态数组： 使用new关键字创建，使用delete关键字释放；

第五章：表达式

操作符优先级： 一元操作符>乘除>加减

逻辑与： *&& 操作符的操作数同时为true才为true；*

逻辑或： || 操作符的操作数一个为true就为true

逻辑非 *： ！ 取相反条件值*

位操作符： 将整型操作数视为二进制位的集合

1.~:位取反

2.<<:左移

3.>>:右移

4.&:位与

5.|：位或

6.^:位异或

条件操作符： cond？expr1：expr2；

```
return s1>s2?s1:s2;
```

自增与自减操作符：++j：先加1，再返回；

j++：先返回，在加1；

箭头操作符：点操作符用于获取类对象成员，箭头用于指向类对象的指针**

sizeof：返回对象或者类型名的长度

注：对指针做$sizeof$操作将返回存放指针所需的大小

内存耗尽异常：bad_alloc

注：删除了指针所指向的对象后，应将指针置0

动态创建的默认初始化：1.类类型对像默认用构造函数；

2.内置类型对象无初始化；

隐式转换：1.混合类型表达式转化为相同类型；

2.条件表达式转化为bool类型；

3.初始化某个变量,转化为该变量类型；

4.函数调用

5.算数转换：在执行算数操作之前转化为同一类型

6.有符号与无符号转换

7.指针转换

8.枚举类型：自动转换为整型

9.const类型：自动将非const转换为const

10.标准库类型定义的转换：如istream到bool的转换

显示转换（强制类型转换）：包括static_cast、dynamic_cast、const_cast、reinterpret_cast

```
cast-name<type>(val)
```

第六章 语句

简单语句：表达式语句

空语句

声明语句：对象与类的定义与声明；

复合语句（块）：一对花括号括起来的语句序列

If语句：判断真假

```
if(....){...}
else{...}
```

switch语句：

```
switch(ch):{
        case‘ *’: ..break;
          ……..
        default:....
}
```

注：标号必须是整型常量表达式

while语句与for语句：条件为真时，反复执行

```
while(...){...}
for(int i=0;i<10;i++){...}
```

do while语句:先执行do，在判断；

```
do
{
    ...
}while(...)
```

break语句：退出当前循环；

continue：当前迭代提前结束；

goto语句：函数内部的无条件跳转；

try、catch、throw：用于异常处理；

```
try{
     ......
}catch(exception){
    ......
}
```

标准异常：1.exception stdexcept new type_info四个标准异常类头文件