2020春季学期C++整理1.0

基本知识点

一.引用的问题:

1.引用是个别名，当建立引用时，程序的另一个变量和对象就名字就初始化它。所以引用其实是作为目标的别名而使用，对引用的改动就是对目标的改动。

基本操作:

```
int a;
int& temp = a;
```

声明temp是对**整数a的引用**，初始化为**引用a**，引用没有值，不占储存空间，声明引用时，目标的储存状态不会改变。**引用只有声明，没有定义**!

对于如下代码:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int intone = 2;
    int& rint = intone;
    cout << intone << endl << rint << endl;
    rint = 7;
    cout << intone << endl << rint << endl;
    cout << &intone << end
    return 0;
}
```

输出结果:

```
2
2
7
7
0x70fe04
0x70fe04
```

引用rint用intone来初始化，以后无论是改变rint的值还是改变intone的值**实际上都是指intone**，二者的值一样,且可以看到**二者地址也相同。**

注意:

（1）.引用在声明时**必须初始化**

（2）.初始化后**一直引用该变量，任何对引用赋值的操作都是对引用所维系的目标的操作**（因为二者地址是一样的）。

（3）.引用的对象只能是变量，而**不能是常量或表达式。**

2.常引用既是在定义引用前加一个 const ， 如下代码:

```
int n = 100;
const int & r = n;
r = 10; //error
n = 10; //accepted
```

其中r既是const int &型的常引用。

其次，**常引用不能用于修改目标的内容**，比如上面的代码，R3会报错但是R4不会。

注意：

（1）**T&型的引用和T类型的变量可以用来初始化const T &类型**的引用。

（2）**const T类型的常变量和const & T 类型的引用则不能用来初始化T&类型的引用，除非强制类型转换**

二.作业题解：

A.

题目描述：

编写主程序，将输入字符串反序输出。程序运行结果如下：

输入ABCDEFGHIJK

输出KJIHGFEDCBA

题解：

考虑字符串的输入输出，第一反应是string。

AC代码：

```
#include<iostream>
#include<cstring>
#include<cstdio>
using namespace std;
int main()
{
    string s;
    cin >> s;
    for (auto i = s.end() - 1; i >= s.begin(); i--) { //string里的指针
        cout << *i;
    }
    cout << endl;
    return 0;
}
```

B.

题目描述：

输入两个整数a,b，使指针变量ip指向变量a，变量和指针方式访问a的值；再将指针变量ip指向变量b，变量和指针方式访问b的值；再将指针变量ip指向变量c，c用于存放a和b之和，变量和指针方式访问c的值。

题解：

标准板子题，利用指针访问不同变量的值。

AC代码：

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<cmath>
#include<algorithm>
#include<vector>
#include<map>
#include<stack>
#include<queue>
using namespace std;
int main()
{
    int* ip;
    int a, b;
    cin >> a >> b;
    ip = &a;
    cout << "a=" << a << endl << "*ip=" << *ip << endl;
    ip = &b;
    cout << "b=" << b << endl << "*ip=" << *ip << endl;
    int c = a + b;
    ip = &c;
    cout << "c=" << c << endl << "*ip=" << *ip << endl;
    return 0;
}
```

C.

题目描述：

通过swap函数交换两个参数的数值，swap函数参数要求为整型指针。

题解：

注意swap函数接受的是指针变量，所以在定义的时候注意！~~其实可以直接全局变量~~

AC代码：

```
#include<iostream>
using namespace std;
void swap(int* a, int* b)
{
    int t;
    t = *a;
    *a = *b;
    *b = t;
}
int main()
{
    int a, b;
    cin >> a >> b;
    cout << "if" << ' ' << "a=" << a << ',' << "b=" << b << endl;
    swap(&a, &b);
    cout << "then" << ' ' << "a=" << a << ',' << "b=" << b << endl;
    return 0;
}
```

D.

课后实验题emmmm.....

代码：

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int icount = 36;
    int *pointer = &icount;
    *pointer = 95;
    cout << icount << endl;
    cout << pointer << endl;
    cout << &icount << endl;
    cout << *pointer << endl;
    cout << &pointer << endl; //双指针
    return 0;
}
```

输出结果：

```
95
0x70fe1c
0x70fe1c
95
0x70fe10
```

三.动态内存分配：

用**new**运算符实现动态内存分配。

1.分配一个变量

```
P = new T
```

动态分配出一片大小为sizeof(T)的内存空间，并将其空间的起始地址分给P

```
int * p;
p = new int;
*p = 5
```

2.分配一个数组

```
int *p;
int i = 5;
p = new int[i*5];
p[0] = 20;
p[100] = 10; //数组越界
```

3.用delete操作归还空间

```
int * p;
p = new int;
delete p;
delete p;//error不能重复删除
```

4.用delete对数组操作

```
int *p = new int [20];
p[0] = 1ll;
delete [] p;//! ! ! ! ! ! ! ! !
```

注意：

对数组进行delete操作时一定要加[]，否则系统只会删除p所指的那片区域——数组的起始位置。而不会去删除整个数组的空间（数组空间连续）