CPP寒假班考试

- 考试时间：120min
- 考试要求：选择题将答案写在题号前面，编程题根据程序的命名要求命名自己的程序，然后完成所需要求，要求完全匹配题目中所规定的输入和输出格式，考试结束后我会将所有人的代码打包然后拷贝到我的U盘上。
- 总分100分，选择题每个8分，大题前两个一个10分，后两个一个20。即便做不出来也要把必要的输入输出写上，我会酌情给分。

选择题

1. 假定x和y为double型，则表达式x=2，y=x+3/2的值是 （）
   A 3.500000
   B 3
   C 2.000000
   D 3.000000
   答案：D， 3/2结果是结果是1，因为/是整数除法，将小数点后抹去，加上x为3.000不是3的原因是y和x都是double
2. 以下关于return语句的描述正确的是（）
   A. 一个自定义函数中必须有一条return语句
   B. 定义成void类型的函数中可以有带返回值的return语句
   C. 一个自定义函数中可以根据不同情况设置多条return语句
   D. 没有return语句的自定义函数在执行结束时不能返回到调用处
   答案：C，A错因为程序返回值为void时，可以不用写return，B错因为返回值void时不可以带返回值的return， D返回值为void的时候即使没有return在执行结束后可以返回调用处
3. 若有语句 `int x = 5, *p = &x;` 则(*p)++相当于（）
   A. x++
   B. p++
   C. *(p++)
   D. *p++
   答案：A， *\p = & x , p指向x，=(*p) ++中的 *p 相当于对p指向的元素取值，对该值++，
4. 有甲乙内三项任务，需2人承担，乙内各需一人承担，从10人中选出4人承担这三项任务不同的选法种数是（）
   A. 1260种
   B. 2025种
   C. 2520种
   D. 5040种
   答案：C首先十个选四个人， 然后四个选两个担任任务甲，然后剩下两个人对乙和丙进行排序，结果是
   $C_{10}^4 \times C_4^2 \times 2$
5. 执行下列的代码，输出的结果是（）

```cpp
#include<iostream>
using namespace std;
int func(int x)
{
    if(x <=4 )
        return 2*x -1 ;
    else if(x >7)
        return func(x-4) +x;
    else
        return func(x+3) +2;
}
int main()
{
    cout<< func(10);
    return 0;
}
```

A. 26
B. 29
C. 38
D. 45
答案：C 递归函数的使用，一步步调用即可

编程题

1. 促销活动
   程序题目：market.cpp 题目描述
   某超市搞促销活动，活动内容：购物金额每满300元 （含300元） 就可以享受“满300减70的优惠。已知小维的购物金额为N（1<Ns1000000），请计算出享受优惠后他需要支付多少元。 例如： N=430，360元 （360=430-1×70） 输入格式
   输入一个正整数N（1<N≤1000000），表示购物金额（单位：元）
   输出格式
   输出一个正整数， 表示享受优惠后需要支付的金额（单位：元）
   样例输入
   430
   样例输出
   360

```cpp
#include<iostream>
using namespace std;

int main()
{
    int num;
    cin >> num;
    cout << num - (num/300)*70;
    return 0;
}
```

2. 大小写转换
   程序题目：format.cpp 要求输入一行字符串，对于其中每个字符，如果是大写字符,需要把它转换成小写字符，如果是小写字符,需要把它转换成大写自负，如果是数字,不做任何处理，然后将处理后的字符串输出
   建议使用
   输入样例：
   3aF9
   输出样例： 3Af9
   输入样例：

7skI2P
输出样例：
7SKi2p

```cpp
#include<iostream>
#include<string>
using namespace std;

int main()
{
    string str;
    cin >> str;
    for (int i = 0; i < str.size(); i++)
    {
        if(str[i] >='a' && str[i] <='z') str[i] = str[i] - 'a' +'A';
        if(str[i] >='A' && str[i] <='Z') str[i] = str[i] - 'A' +'a';
    }
    cout << str;
    return 0;
}
```

3. 第三大的数 程序题目：findmax.cpp 问题描述：
请你找出一个数组中第三大的数字是多少，如果不存在，则返回数组中最大的数。程序输入分为两个部分，第一个部分是本次输入的数字个数，第二部分是本次输入的数字。输出为一个数，为第三大的数字或者最大值。程序没有最大的数分为两个情况，第一个情况是本次输入的数字小于三个，或者本次没有第三大的数字。
输入格式： 第一行一个整数n表示本次输入几个数字 （n>0, n <10000） 第二行n个整数，表示要在这n个整数中找出第三大的数。 (保证所有测试样例中n是大于0的) 样例如下： 程序输入： 3
3 2 1
程序输出： 1 解释： 程序中第三大的数字为1

输入： 2 1 2
输出： 2
解释：程序中只有两个不同数第三大的数不存在, 所以返回最大的数 2 。

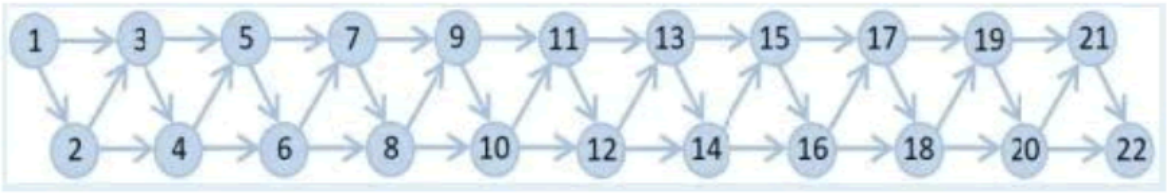
程序输入： 4
2 2 3 1 输出：
1 解释：注意，要求返回第三大的数，是指在所有不同数字中排第三大的数。 此例中存在两个值为 2 的数，它们都排第二。在所有不同数字中排第三大的数为 1 。

```cpp
#include<iostream>
#include<algorithm>
using namespace std;
int arr[10001];

int main()
{
    int num;
    cin>> num;
    for (int i = 1; i <=num; i++)
    {
        cin >> arr[i];
    }
    sort(arr+1,arr+num+1);
    int n_max = 1;
    for (int i = num; i >=2; i--)
    {
        if(arr[i-1] < arr[i]) n_max ++;
        if(n_max == 3){
            cout<< arr[i-1];
            return 0;
        }
    }
    cout << arr[num];

    return 0;
}
```

4. 找路线
   程序题目：path.cpp 题目描述：
   现有22名小朋友，依次编号1到22，22名小朋友分别按照下图的位置站好。



   每名小朋友只能按照图中箭头指向的方向移动。给出两名小朋友的编号N和M（1N<M≤22），请你找出从编号N到编号M共有多少条不同的路线。
   例如：N=3，M=7，从编号3的位置到编号7的位置共有5条路线，分别为：（3->5->7），（（3->5->6->7），（3->4->5->7），（3->4->5->6->7），（3->4>6->7）
   输入格式：
   输入两个正整数N和M（1≤N<M≤22），分别表示两名小朋友的编号，之间以一个空格隔开
   输出格式：
   输出一个整数，表示从编号N到编号M共有多少条不同的路线。
   输入样例：
   3 7 输出样例： 5
   输入样例： 1 3 输出样例：
   2

分析：可以看到3-7的路径数目与1-5的路径数目是相同的，因为他们的拓扑结构相同，并且6-10的路径数目和2-6的路径数目相同，因为他们的拓扑结构相同，所以对于一个任意对的起始点和终点我们可以将它们移动到头部进行分析。并且，起始点为1/2分析是不同的，所以将它们分开分析。3->7的路径数目与1->5 ,并且1-5 与5-1路径相同只是逆序罢了，5-1 路径数目等于(5-4,4-1)的路径和加上(5-3,3-1)的路径和。于是我们可以使用一个递归的算法实现它，递归函数为f(n),f(n) = f(n-1) + f(n-2) ,这时候我们需要考虑递归终止条件了，首先考虑起点为1的情况，起点为1的时候，f(1) f(2) 都是1，才可以保证接下来的顺利推导。同理可以得到起始点为2的情况。 代码示例：

```cpp
#include<iostream>
using namespace std;
int calculate_path_1(int x )  // 起始点为1的情况
{
    if(x==1 || x ==2) return 1;  //终止条件
    return calculate_path_1(x-1) + calculate_path_1(x-2);

}
int calculate_path_2(int x ) // 起始点为2的情况
{
    if(x==2||x==3) return 1; //终止条件
    return calculate_path_2(x-1) + calculate_path_2(x-2);

}
int main()
{
    int n,m,x;
    cin >> n >> m;
    if(n %2 == 1) cout<< calculate_path_1(m-n+1); //如果起始点为奇数可以将它等价到1到m-n+1的路径和
    if(n %2 == 0) cout<< calculate_path_2(m-n+2);//如果起始点为偶数可以将它等价到2到m-n+2的路径和
    return 0;
}
```