# 实验 循环结构

樊子谦

软件学院 软件工程 2024级1班

【1】

编写单条C++语句，完成下述任务

a) 输入整型变量x；

b) 输入整型变量y；

c) 设置整型变量i为1；

d) 设置整型变量power为1；

e) 变量power乘以变量x后，结果赋值给变量power；

f) 变量i后置方式自增1；

g) 判断变量i是否小于或等于变量y；

h) 输出变量power。

按部就班中译C++即可。

#include <bits/stdc++.h>

using namespace std;

int main() {

int x, y;

cin >> x;

cin >> y;

int i = 1;

int power = 1;

power \*= x;

i++;

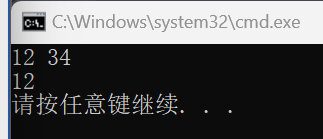
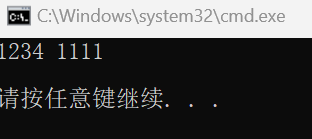
if (x <= y)

cout << power;

return 0;

}

运行结果：



【2】

利用任务【1】的语句，编写计算x的y次幂的程序，要求使用while循环语句。

可以使用非递归快速幂优化至log(y)的复杂度，但是没有必要。直接循环，把初始化为1的临时变量乘y次x即可。

#include <bits/stdc++.h>

using namespace std;

int main() {

int x, y;

cin >> x;

cin >> y;

int i = 1;

int power = 1;

while (1) {

power \*= x;

i++;

if (i <= y)

continue;

else {

cout << power;

break;

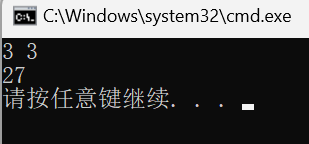
}

}

return 0;

}

运行结果：



（3的3次方为27）

【3】

在计算机应用中，我们常常会遇到寻找最大数的问题。请通过循环语句判定和打印用户输入的10个数中的最大数。你的程序应用到下面3个变量：

counter：计数器，用于记录已输入数的个数；

number：当前输入到程序的数；

largest：截至当前输入已经找到的最大数。

同样没有什么需要注意的地方。直接实现。

#include <bits/stdc++.h>

using namespace std;

int main() {

int counter = 0;

int number;

int max\_num = INT\_MIN;

while (counter < 10) {

cout << "Enter a number: ";

cin >> number;

max\_num = max(max\_num, number);

counter++;

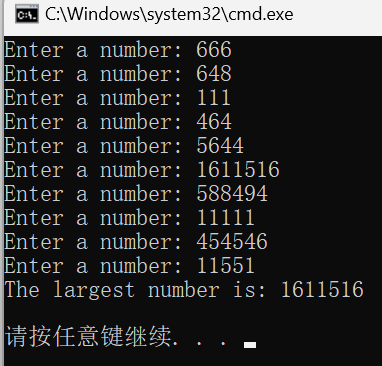
}

cout << "The largest number is: " << max\_num << '\n';

return 0;

}

运行结果：



【4】

用类似任务【3】的方法，从10个数中寻找最大的数和最小的数。并解决歌星大奖赛问题。在歌星大奖赛中，有10个评委为参赛的选手打分，分数为1~100分。选手最后得分为：去掉一个最高分和一个最低分后其余8个分数的平均值。请编写一个程序实现。

在上一题的基础上，记录最小值和总和。最后得分为＝（总和－最低分－最高分）/8，注意转化为浮点数运算。

#include <bits/stdc++.h>

using namespace std;

int main() {

int counter = 0;

int number;

int max\_num = INT\_MIN, min\_num = INT\_MAX, sum = 0;

while (counter < 10) {

cout << "Enter a number: ";

cin >> number;

max\_num = max(max\_num, number);

min\_num = min(min\_num, number);

sum += number;

counter++;

}

double average = 1.0 \* (sum - min\_num - max\_num) / 8;

cout << "The largest number is: " << max\_num << '\n';

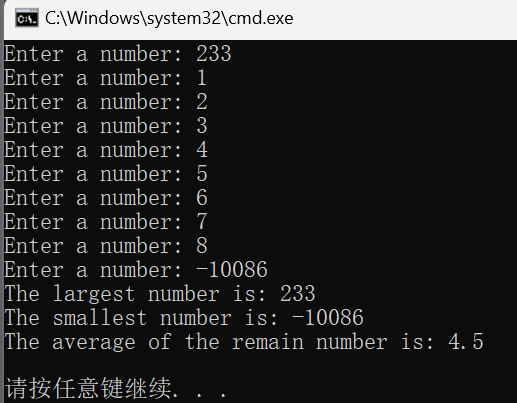
cout << "The smallest number is: " << min\_num << '\n';

cout << "The average of the remain number is: " << average << '\n';

return 0;

}

运行结果：



【6】

中国有句俗语叫“三天打鱼两天晒网”。某人从1990年1月1日起开始“三天打鱼两天晒网”，问这个人在以后的某一天中是“打鱼”还是“晒网”。 （选做）

很麻烦的题目。

这里用具有可扩展性的算法来实现，不默认1990年1月1日为初始日期，把初始日期记作start，查询的某一天为end。实现一个功能daysBetweenDate计算它们之间的日子days:

days=从start年的第一天到end年的前一年最后一天的天数－start年第一天到start那一天的天数+end年到end那一天的天数

可能有很多种表达方式，但是只有这种表达方式能最好地处理start和end在同一年的情况。在同一年时，程序的处理会将“从start年的第一天到end年的前一年最后一天的天数”认为是0.

然后循环遍历，对一整年的天数情况判断闰年；对每一个月的情况，用一个函数返回每个月的天数（2月份做判断闰年的特殊处理）。然后有条理地相加相减即可。还要考虑到用户输入错误（日子在1990.1.1之前或者日期输入格式不对）的问题。这里我们引入sscanf来判断格式问题。

然后就很简单了，因为是5天为一个周期，那么“三天打鱼”就是模5余0、1、2的，“两天晒网”就是余3、4的。

#include <iostream>

using namespace std;

bool isLeapYear(int year) {

if (year % 400 == 0)

return true;

if (year % 100 == 0)

return false;

if (year % 4 == 0)

return true;

return false;

}

int daysInMonth(int year, int month) {

int days[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

if (month == 2 && isLeapYear(year))

return 29;

return days[month - 1];

}

bool isValidDateFormat(const char\* date) {

int year = 0, month = 0, day = 0;

if (sscanf(date, "%d/%d/%d", &year, &month, &day) != 3)

return false;

if (year < 1919 || month < 1 || month > 12 || day < 1 || day > daysInMonth(year, month))

return false;

return true;

}

int daysBetweenDates(char date1[], char date2[]) {

int year1, month1, day1;

int year2, month2, day2;

if (!isValidDateFormat(date1) || sscanf(date1, "%d/%d/%d", &year1, &month1, &day1) != 3)

return -1; // Invalid date1 format

if (!isValidDateFormat(date2) || sscanf(date2, "%d/%d/%d", &year2, &month2, &day2) != 3)

return -1; // Invalid date2 format

cout << year1 << month1 << day1 << endl;

cout << year2 << month2 << day2 << endl;

int days = 0;

for (int year = year1; year < year2; year++) {

days += isLeapYear(year) ? 366 : 365;

}

for (int month = 1; month < month1; month++) {

days -= daysInMonth(year1, month);

}

for (int month = 1; month < month2; month++) {

days += daysInMonth(year2, month);

}

days += day2 - day1;

return days;

}

int main() {

char date1[] = "1919/10/10";

char date2[12];

int days;

cout << "The first date is 1919/10/10." << '\n';

cout << "Enter date2 in the format (yyyy/mm/dd):" << '\n';

while (1) {

cin >> date2;

days = daysBetweenDates(date1, date2);

if (days < 0)

cout << "Invalid date format. Please make sure you enter date2 in the format (yyyy/mm/dd) and this day is NOT before 1919/10/10." << endl;

else

break;

}

cout << "Number of days between " << date1 << " and " << date2 << ": " << days << '\n';

int remainders = days % 5;

if (remainders <= 2)

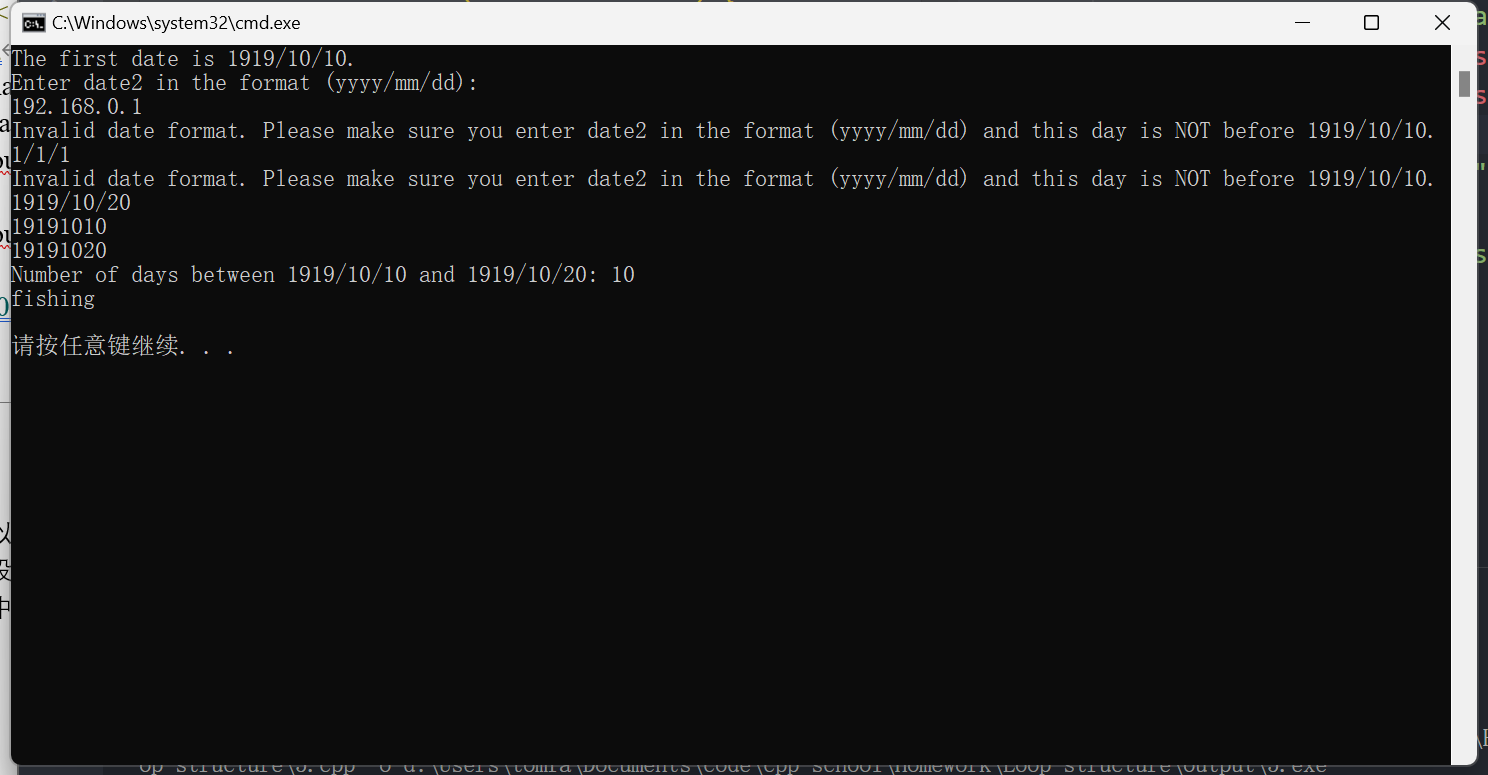
cout << "fishing" << '\n';

else

cout << "dry nets" << '\n';

return 0;

}

输出：  


明显看到程序对错误的输入是有纠正作用的。

心得：

使用循环可以完成很多具有重复性和相同性的任务，循环结构是一个很重要的结构。

这次实验并没有要求数据范围，所以像【2】这样的容易溢出的问题也没有特殊处理，但是在实际工程中也是要注意的。

像【6】这种任务，题面没有给输入格式，就要自己去规定输入格式，但是这就可能产生非法输入引起程序崩溃的问题。这里使用了sscanf简单处理，如果使用cin，处理将会更加复杂。这种处理在实验《类与对象》有出现，对设计程序的严谨性有很强的要求。