上海财经大学《程序设计基础》期末考试卷(A 卷)

（ 2022 至 2023 学年 第 1 学期 ）

【考试形式】 开卷：允许携带各种纸质资料 ，不允许上网查资料，不允许查阅手机、平板电脑和笔记本电脑等具有通讯功能的设备上的资料。

【答题要求】
（1）请将答案写在答题纸指定位置。
（2）答题之前，在答题纸第 1 页抄写诚信承诺并签名，每一页上填写”姓名、学号和班级”信息。
（3）考试期间全程开启摄像头，摄像头能够拍摄到面部与手部。
（4）考试过程中有问题，可以通过腾讯会议聊天提问。
（5）原则上学生不可以提前交卷；当教师宣布结束考试时，不允许再做答。
（6）在考试时间结束的 10 分钟内：首先拍摄答题纸，确保清晰度；然后将答题纸照片按页码顺序插入答卷 word 文档，答卷文档命名规则：学号+姓名+程序设计基础答卷。如：2022 110814_张三_程序设计基础答卷.docx 或.pdf。将你的答卷 word 或 pdf 文档提交至 BB 系统，在教师确认收到答卷文档后，方可离开考场。

1. 循环控制（编程题，15 分）

编写程序计算三角函数 sin*x* 近似值。请根据 sin*x* 泰勒展开式，应用循环编写程序。

$$\sin x = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} x^{2k+1} = x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 - \frac{1}{7!}x^7 + \cdots$$

当 $\left|\frac{(-1)^k}{(2k+1)!}x^{2k+1}\right| < 10^{-9}$ 时，循环结束。

程序如下：

```
#include <iostream>
using namespace std;
int main()
{
    double x, sinX = 0;
    cout << "Please input the value of x ): ";
    cin >>x;
    //请补充完成以下缺少的代码




    cout << "sin(" << x << ") = " << sinX << endl;
    return 0;
}
```

2. 函数调用（编程题，20 分）

编写名为 calculateStatistics 的函数，实现如下功能：计算整型数组元素中的平均值、方差，以及数组中大于等于平均值的元素个数和小于平均值的元素个数。假设数组 x，包含 N 个元素:

数组元素平均值： $x\_mean = \sum_{i=1}^{N} x_i$

数组元素的方差： $x\_deviation = \frac{1}{N}\sum_{i=1}^{N}(x_i - x_{mean})^2$

```
// 主程序
#include <iostream>
using namespace std;
const int NUM = 10;
int main()
{
   int x[NUM] = {12, 23, 41, 28, 19, 27, 38, 14, 52, 32};
   int greaterNum = 0, lessNum = 0;
   double x_mean =0, x_deviation = 0;
   x_mean = calculateStatistics(x, NUM, &x_deviation, greaterNum, lessNum);
   cout <<"数组元素平均值: " << x_mean << endl
        <<"数组元素的方差: " << x_deviation << endl
        <<"大于等于平均值的元素: "<< greaterNum << "个" << endl
        <<"小于平均值元素:"<< lessNum << "个" <<endl;
   return 0;
}

//请编写 calculateStatistics 的函数
```

3. 数组与字符串（改错题，20 分）

本题目的程序可以对输入的 3 个句子(包含空格的字符串)进行升序排序。并输出排好序的句子。程序正常运行示例如下：

```
Please input three sentences:
The Chinese people are hardworking people!
The Chinese people are selfless and enterprising people!
The Chinese people are brave to innovate!

Sorted sentences:
The Chinese people are brave to innovate!
The Chinese people are hardworking people!
The Chinese people are selfless and enterprising!
```

下面是实现上述功能的程序，请仔细阅读程序，找出其中错误并修改。

注意：

1）不能改变 main 函数中的数据定义：char sentences[MAXNUM][MAXLENGHT];

2）不能改变 stringSort 函数中的选择排序算法。

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  const int MAXNUM = 3;
6  const int MAXLENGHT = 80;
7  void stringSort(char str[MAXNUM][], int n){
8      char *temp;
9      for(int i=0; i<MAXNUM-1; i++){
10         k = i;
11         for(int j=i+1; j<MAXNUM; j++)
12             if(strcmp(str[k],str[j])) k = j;
13         if(k!=i){
14             temp = str[i];
15             str[i] = str[k];
16             str[k] = temp;
17         }
18     }
19 }
20
21 int main(){
22     char sentences[MAXNUM][MAXLENGHT];
23     cout << "Please input three sentences:" << endl;
24     for(int i=0; i<MAXNUM; i++)
25         cin >> sentences[i] << endl;
26     cout << endl;
27     stringSort(sentences, MAXNUM);
28     cout << "Sorted sentences: " << endl;
29     for(int i=0; i<MAXNUM; i++)
30         cout << sentences[i] << endl;
31     return 0;
32 }
```

4. 指针与动态数组（改错题，17 分）

下面程序功能包括：

1） 将 3 个字符串，每个字符串去除其中非数字和非英文字母的字符，并将小写字母转化为大写字母，然后进行连接合并；

2） 统计 3 个字符串中数字字符和英文字母的频次；

3） 输出连接合并后的字符串、以及 3 个字符串中数字字符和英文字母的频次。

例如：const char *str[3] = { "I love VC++ 2010!",
"Do you like Python 3.7?",
"Java SE18 is good!"
};

去除非英文字母和非数字的字符  小写字母转化为大写字母，进行连接合并

mergestring：ILOVEVC2010DOYOULIKEPYTHON37JAVASE18ISGOOD
中英文字母和数字字符的频次：alphabet[ ]，digital[ ]

输出连接合并串  英文字母和数字字符的频次

Merged strings: ILOVEVC2010DOYOULIKEPYTHON37JAVASE18ISGOOD
数字字符的频次：0:2 1:2 2:1 3:1 7:1 8:1
英文字母的频次：A:2 C:1 D:2 E:3 G:1 H:1 I:3 J:1 K:1 L:2 N:1 O:6 P:1 S:2 T:1 U:1 V:3 Y:2

仔细阅读实现上述功能的 MergeStatistics 函数的实现，找出其中的错误，并（在不改变程序结构基础上）进行修改。

提示：注意用于计算频次的数组下标、大小写转换、内存空间的动态分配等。

```
1  #include <iostream>
2  #include <cstring>
3  #include <cctype>
4  using namespace std;
5  const int MAXLENGTH = 3;
6  //去除非英文字母和非数字的字符，小写字母转化为大写字母，返回连接合并的字符串指针
7  char *MergeStatistics(const char *str[], int MaxLen, int digital[], int alphabet[]){
8      int mergedlen = 0;
9      for(int i=0; i<MaxLen; i++)
10         mergedlen += strlen(str[i]);
11     char *mergestring=new char(mergedlen+1);
12     int counter = 0;
13     for(int i=0; i<MaxLen; i++){
14         const char *ptr = str[i];
15         char c;
16         while((c = *ptr)!='/0'){
17             if(c>='A'&&c<='Z'|| c>='a'&&c<='z'){
18                 c = c -32;
```

```
                ++alphabet[c];
                mergestring[++counter] = c;
            }
            if(c>='0'&&c<='9'){
                ++digital[c];
                mergestring[++counter] = c;
            }
            ptr++;
        }
    }
    return mergestring;
}
// 以下主程序无须修改
int main(){
    const char *str[MAXLENGTH] = { "I love VC++ 2010!",
                                   "Do you like Python 3.7?",
                                   "Java SE18 is good!"
                                 };
    int digital[10] = {0};      // 存放数字字符频次的数组，
    int alphabet[26] = {0};   // 存放英文字母频次的数组
//去除非英文字母和非数字的字符，小写字母转化为大写字母，返回连接合并的字符串指针
    char *mergedStr = MergeStatistics(str, MAXLENGTH, digital, alphabet);
    cout << "Merged strings: " << mergedStr << endl;   // 输出连接合并的字符串
    for(int i=0; i<10; i++)     // 输出数字字符的频次
        if(digital[i]) cout << i << ":" << digital[i] <<"    ";
    cout << endl;
    for(int i=0; i<26; i++)   // 输出英文字母的频次
        if(alphabet[i]) cout << char(i+'A') << ":" << alphabet[i] <<" ";
    delete []mergedStr;       // 释放 new 申请的空间
    return 0;
}
```

5. 递归（编程与问答，10 分）

两个非负整数的最大公约数（经常被简写为 gcd）就是能整除这两个整数的因数中最大的。

希腊数学家欧几里得发现两个非负整数 x, y 的最大公约数可以用如下的方法计算：

（I）如果 x 可以被 y 整除，则 y 就是最大公约数；

（II）否则，x 和 y 的最大公约数总是等于 y 和 y 除 x 所得的余数的最大公约数。也即：

gcd(x,y) === gcd(y, y 除 x 所得的余数)

（1）利用欧几里得的上述发现，编写一个递归函数 gcd（x，y），计算 x 和 y 的最大公约数。

```
int gcd(int x, int y)
{
//请补充完成以下缺少的代码




}
```

（2）你编写的上述递归程序是否总是能正常结束？请分析并解释原因。

6. 类（编程题，18 分）

校园附近开张了一些汉堡连锁店。每个店铺的点餐系统都用动态数组对汉堡进行管理。其中汉堡是结构体 Burger，采用接口（.h 文件）与实现（.cpp 文件）分离的多文件方法，创建一个汉堡商店类 BurgerShop，具体方法是用一个动态数组（由 burger_arr 指针指向该动态数组）管理汉堡店中的所有汉堡。该动态数组以 5 个汉堡结构体为单位分配空间。例如：当汉堡店的汉堡从 5 增加到 6 时，动态数组空间从可以记录 5 个 Burger 扩充到可以记录 10 个 Burger；汉堡店每销售一个汉堡，从数组中删除该汉堡信息，并将汉堡总数 burger_num 减一（为了简单，卖出汉堡时，并不减少存储空间）。

请实现 BurgerShop 类中的拷贝构造函数、重载赋值操作符、重载 << 操作符、新增汉堡的 AddBurger 函数，补充其中缺少的代码。

//BurgerShop.h（类 BurgerShop 的头文件）

```
#ifndef __BURGERSHOP_H
#define __BURGERSHOP_H
#define ALLOC_SPACE 5
#define MAX_PRICE 30
#define MAX_NUTRIENT 10

#include <iostream>
using namespace std;
struct Burger
{
    int price;
    int nutrient;
};
```

```
class BurgerShop
{
public:
    BurgerShop();                                                    //构造函数
    BurgerShop(const BurgerShop& burgerShop);                        //拷贝构造函数
    void AddBurger(Burger burger);                                   //新增一个汉堡
    BurgerShop& operator= (const BurgerShop& burgerShop); //重载赋值操作符
    friend ostream & operator<<(ostream & os, const BurgerShop bp);   //重载 << 操作符
    ~BurgerShop();                                                   //析构函数
    void SellBurger(int index);                                      //卖出一个汉堡
    Burger & operator[ ](int index);     //重载 [] 操作符
private:
    Burger* burger_arr;
    int burger_num;
};
#endif //__BURGERSHOP_H
```

// BurgerShop.cpp（类 BurgerShop 的实现文件）

```
#include <stdlib.h>
#include <cstring>
#include <cmath>
#include <iostream>
#include "BurgerShop.h"
using namespace std;

BurgerShop::BurgerShop()
{
    burger_arr = NULL;
    burger_num = 0;
}

BurgerShop::BurgerShop(const BurgerShop & burgerShop)
{
  // 请补充缺少以下的代码 6.1




}

void BurgerShop::AddBurger(Burger burger)
{
    // 请补充缺少以下的代码 6.2
```

}

// 注： 调用 cout << burgerShopA; 在屏幕上的输出的示例如下，请据此来编写 operator<< 重载函数。

```
burger number: 8
burger 1-> price: 28 nutrient: 0
burger 2-> price: 15 nutrient: 2
burger 3-> price: 20 nutrient: 7
burger 4-> price: 21 nutrient: 4
burger 5-> price: 11 nutrient: 2
burger 6-> price: 23 nutrient: 4
burger 7-> price: 14 nutrient: 9
burger 8-> price: 19 nutrient: 2
```

```
ostream & operator<<(ostream & os, const BurgerShop bp){
    // 请补充缺少以下的代码 6.3



}

BurgerShop& BurgerShop::operator=(const BurgerShop &burgerShop)
{
    // 请补充缺少以下的代码 6.4



}

void BurgerShop::SellBurger(int index)
{
    if (index>burger_num II index < 1)
    {
        cout<<"no such burger\n";
        return ;
    }
    for(int i=index-1;i<burger_num-1;i++)
        burger_arr[i] = burger_arr[i+1];
```

```
    burger_num--;
}

Burger & BurgerShop::operator[ ](int index){
    return burger_arr[index];
}

BurgerShop::~BurgerShop()
{
    if (burger_arr!=NULL)
        delete [] burger_arr;
}
```

// main.cpp（主程序文件，以下程序仅说明类中相关函数的使用场景，答题时并不需要）

```
#include <iostream>
#include <ctime>
#include <stdlib.h>
#include "BurgerShop.h"
using namespace std;

int main()
{
    srand(time(NULL));
    BurgerShop burgerShopA;     // 此处调用默认构造函数
    //（1）以下首先为 burgerShopA 汉堡店准备若干汉堡，随机生成每个汉堡的数据（价格、营养值）。
    for(int i=1;i<=8;i++)
    {
        int tmp = rand();
        Burger tmp_burger = { tmp%MAX_PRICE+1, tmp%(MAX_NUTRIENT+1)};
        burgerShopA.AddBurger(tmp_burger);     // 此处调用 AddBurger 函数
    }
    //（2）显示 burgerShopA 汉堡店中的每一个汉堡
    cout << burgerShopA <<endl;     // 此处调用重载的 operator<<函数
    //  （3）构造一个新的汉堡店 burgerShopB，复制 burgerShopA 的所有汉堡。
    BurgerShop burgerShopB=burgerShopA;   // 此处调用拷贝构造函数
    //  （4）burgerShopA 销售一个汉堡：选择汉堡、显示选择、更新数据、显示销售后汉堡店中剩余的汉堡。
    int no;
    cout << "which one do you like to buy?"<< endl;
    cin >> no;
    cout << "you want to buy :" << "(" << burgerShopA[no-1].price<<","
         <<   burgerShopA[no-1].nutrient << ")"<<endl;        // 此处调用重载的 operator[] 函数
    burgerShopA.SellBurger(no);     // 此处调用 SellBurger 函数

    cout << "The burgers in burgerShopA After the sell:"<< endl
         << burgerShopA <<endl;              // 此处调用重载的 operator<<函数
```

```
    cout << "The burgers in burgerShopB :"<< endl << burgerShopB <<endl;   // 此处调用重载的 operator<<函数

    //（5）交换 burgerShopA 与 burgerShopB
    //首先调用拷贝构造函数，创建一个临时对象 burgerShopX；然后赋值（调用赋值操作符）。
    BurgerShop burgerShopX=burgerShopA;
    burgerShopA=burgerShopB;
    burgerShopB=burgerShopX;
    cout << "After interchange:" <<endl
         << "The burgerShopA:"<< endl << burgerShopA <<endl;
    cout << "The burgerShopB :"<< endl << burgerShopB <<endl;

    return 0;
}
```