Group 3 Assignment 2

组员编号	姓名	学号
1	张晟赫	D24091111148
2	罗琢	D24091110572
3	王国霖	D24091111111
4	孙梦娜	D24091110375

作业要求：1）以小组形式完成作业，写上小组人员姓名及学号；

2）在下面文档对应题后补全代码，并合理注释；

3）运行程序设计的代码，给出示例，并录屏上传至畅课。

- 选择题（共20分）

1．已知 char *a[]={ "fortran", " basic", "pascal","java", "c++" }; 则 cout<<a[3];的显示结果是（ C ）。

（A）t　　（B）一个地址值　　（C）java　　（D）javac++

2．设有 char *s="ABCDE"; cout<<*(s+1)<<endl; 输出结果是（ B ）。

（A）A　　（B）B　　（C）ABCD　　（D）BCD

3．设有 char *s="ABCDE"; cout<<(s+1)<<endl; 输出结果是（ D ）。

（A）A　　（B）B　　（C）ABCD　　（D）BCDE

4．设有 char *s="ABCDE"; cout<<strlen(s)<<endl; 输出结果是（ B ）。

（A）6　　（B）5　　（C）4　　（D）1

5．设 char *s1, *s2; 分别指向两个字符串，可以判断字符串s1和s2是否相等的表达式为（ D ）。

（A）s1=s2　　（B）s1==s2

（C）strcpy(s1,s2)==0　　（D）strcmp(s1,s2)==0

- 程序设计（共80分）

1. 使用指针函数编写程序，把两个随机输入的字符串连接起来。要求不能使用已有库函数。（15分）

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

// 连接两个字符串的函数
void connect(char *dest, const char *str1, const char *str2) {
    // 定义指针指向dest
    char *ptr = dest;

    // 复制第一个字符串 用 while 遍历字符串 遇到结束符结束遍历
    while (*str1 != '\0') {
        *ptr = *str1;
        ptr = ptr + 1 ;
        str1 = str1 + 1;
    }

    // 复制第二个字符串 同理
    while (*str2 != '\0') {
        *ptr = *str2;
        ptr = ptr + 1 ;
        str2 = str2 + 1;
    }
```

```
    // 添加字符串结束符
    *ptr = '\0';
}

int main() {
    // 定义最大长度
    const int MAX_SIZE = 100;
    // 定义字符串
    char str1[MAX_SIZE];
    char str2[MAX_SIZE];
    // 存储结果
    char result[2 * MAX_SIZE];

    // 输入字符串
    cout << "输入拼接左边的字符串:";
    cin.getline(str1, MAX_SIZE);

    cout << "输入拼接右边的字符串: ";
    cin.getline(str2, MAX_SIZE);

    // 调用函数连接字符串
    connect(result, str1, str2);

    cout << "拼接后的字符串: " << result << endl;

    return 0;
}
```

2. 随机输入10个整数，使用带指针参数的函数实现按输入顺序的逆序排列并输出。（15分）

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

// 使用指针逆序排列数组函数
void reverseArray(int *arr, int size) {
    // 定义指针
    int *left = arr;
    int *right = arr + size - 1;

    // 双指针遍历交换
    while (left < right) {
        // 交换元素

        int temp = *left;
        *left = *right;
        *right = temp;

        // 左指针右移
        left++;
        // 右指针左移
        right--;
    }
}

int main() {
    // 定义最大长度
    const int SIZE = 10;
    // 定义函数
```

```
    int numbers[SIZE];

    // 随机输入10个整数
    cout << "输入10个整数: " << endl;
    for (int i = 0; i < SIZE; i++) {
        cin >> numbers[i];
    }

    // 调用函数输出答案
    reverseArray(numbers, SIZE);

    // 输出答案
    cout << "逆序后的数组: " << endl;
    // 遍历输出
    for (int i = 0; i < SIZE; i++) {
        cout << numbers[i] << " ";
    }
    cout << endl;

    return 0;
}
```

3. 按照指定长度生成动态数组，用随机数对数组元素进行赋值，然后逆置该数组元素。例如，某个数组的初值为{5, 4, 3, 8, 2}，逆置后的值为{2, 8, 3, 4, 5}。要求输出逆置前、后的数组元素序列。（25分）

```
#include <iostream> // 用于输入输出

using namespace std;

// 定义随机数种子
unsigned int seed = 1;

// 定义随机数生成器函数
unsigned int my_rand()
{

    seed = seed * 1103515245 + 12345;
    // 返回 rand
    return (unsigned int)(seed / 65536) % 32768;
}

int main()
{
    int n;
    cout << "输入数组的长度: ";
    cin >> n;

    // 定义数组
    int *arr = new int[n];

    cout << "请输入一个随机数种子（正整数）: ";
    cin >> seed;

    // 随机数赋值数组
    cout << "逆置前的数组元素为: ";
    for (int i = 0; i < n; i++)
    {
        arr[i] = my_rand() % 100;
```

```
        // 生成随机数
        cout << arr[i] << " ";
    }
    cout << endl;

    // 逆置数组元素
    for (int i = 0; i < n / 2; i++)
    {
        // 利用中间变量进行swap
        int temp = arr[i];
        arr[i] = arr[n - 1 - i];
        arr[n - 1 - i] = temp;
    }

    // 输出逆置后的数组
    cout << "逆置后的数组元素为: ";
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    cout << endl;

    // 释放动态分配的内存
    delete[] arr;

    return 0;
}
```

4. 参照以下代码，补全两个函数以完成对字符串的插入和删除操作。其中两个函数实现形式为：

5.

insertStr(text,s,n); //在字符串的第n个字符后插入s串

deleteStr(text,start,n); //删除字符串中从第start 个字符开始，连续n个字符的串

注意：函数不需要考虑字符串的允许长度。请补全以上定义的insertStr和deleteStr函数中的代码块，并给出代码使用示例。要求不得使用标准库函数。（25分）

```
#include<iostream>

using namespacestd;

**void insertStr(char *t, char *s, int n);**

**void deleteStr(char *t, int start, int n);**

void main()

{    char text[256]="\0";

char s[128]="\0";

int k,n,start;

while(1) {

cout<<"当前字符串为: "<<text<<endl;
```

```
cout<<"选项: 1-插入字符串  2-删除字符串  0-退出\n";

cin>>k;

switch(k) {

case 1:

{  cout<<"输入需要插入的字符串: ";

cin>>s;

cout<<"输入插入字符串的位置";

cin>>n;

         **insertStr(text,s,n)**;

break;

}

case 2:

{  cout<<"输入删除字符串开始位置: ";

cin>>start;

cout<<"输入被删字符串长度";

cin>>n;

**deleteStr(text,start,n)**;

break;

}

case 0: return;

}

}

}
```

//补充函数定义

```
void insertStr(char *t, char *s, int n)
{
    // 计算字符串的长度
    int len_t = 0;
    while (t[len_t] != '\0')
        len_t++;

    int len_s = 0;
    while (s[len_s] != '\0')
        len_s++;
```

```
    // 计算插入位置的索引
    int insert_pos = n;

    // 将原字符串中从插入位置开始的字符向后移动
    for (int i = len_t; i >= insert_pos; i--)
    {
        t[i + len_s] = t[i];
    }

    // 遍历复制字符串
    for (int i = 0; i < len_s; i++)
    {
        t[insert_pos + i] = s[i];
    }
}
```

```
void deleteStr(char *t, int start, int n)
{
    // 计算字符串长度
    int len_t = 0;
    while (t[len_t] != '\0')
        len_t++;

    // 计算插入位置的索引
    int del_start = start - 1;

    // 左移字符串
    for (int i = del_start + n; i <= len_t; i++)
    {
        t[i - n] = t[i];
    }
}
```