电子信息与工程学院

姓名	学号	专业	联系方式
黄炜恺	17311023	微电子科学与工程	13725951349

第一题

代码

```
#include <iostream>
#include <string.h>

using namespace std;

const int maxqueue = 10;
enum Error_code{success,underflow,overflow};

template <typename  T> class Extended_queue {
 public:
   Extended_queue();
  bool empty() const;
  Error_code serve();
  Error_code append(const T &item);
  Error_code retrieve(T &item) const;

  bool full() const;
  int size() const;
  void clear();
  Error_code serve_and_retrieve(T&item);

 protected:
  int count;
  int front;
  int rear;
  T entry[maxqueue];
};

template <typename T>
bool Extended_queue<T>::full()const
{
    return (count == maxqueue) ? true : false;
}

template <typename T>
int Extended_queue<T>::size()const
{
    return count;
}

```

```cpp
template <typename T>
void Extended_queue<T>::clear()
{
    for(int i=front ; i < front+count ; i++)
        delete &entry[i];
    count = 0;
    front = 0 ;
    rear = 0;
}

template <typename T>
Error_code Extended_queue<T>::serve_and_retrieve(T&item)
{
    if (count == 0)
    return underflow;
    else
    {
        item = entry[front];
        delete &entry[front];
        return success;
    }
}

//初始化
template <typename T> Extended_queue <T>::Extended_queue()
{
    count = 0;
    front = 0 ;
    rear = 0;
}

//检查队是否空
template <typename T> bool Extended_queue <T> ::empty() const
{
    return (count == 0) ? true : false;
}
```

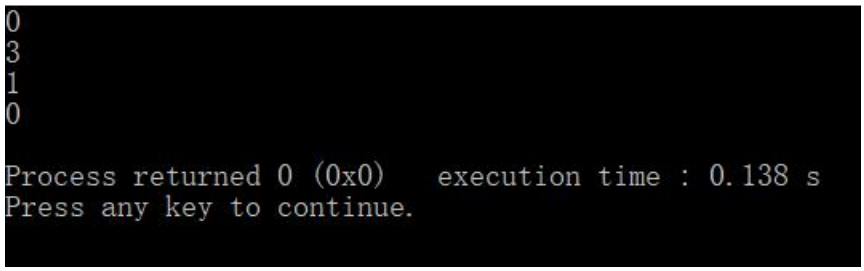
```cpp
//出队
template <typename T> Error_code Extended_queue <T> :: serve ()
{
    if (count)
    {
        delete &entry[front] ;
        front = ((front == maxqueue-1) ? 0 : (front + 1));
        return success;
    }
    else
        return underflow;
}

//队尾插入一个元素
template <typename T> Error_code Extended_queue <T> :: append (const T&item)
{
    if(count >= maxqueue)
        return overflow;
    count++;
    entry [rear] = item;
    rear = ((rear+1) == maxqueue) ? 0 : (rear+1);
    return success;
}

//取队头元素
template <typename T> Error_code Extended_queue <T> :: retrieve(T&item) const
{
    if (count == 0)
        return underflow;
    item = entry[front];
    return success;
}
```

```
class Stu
{
    char Name[20];  //学生姓名
    float Chinese;      //语文成绩
    float Math;     //数学成绩
public:
    float Average(void);    //计算平均成绩
    float Sum(void);    //计算总分
    void Show(void);    //打印信息
    void SetStudent(char*,float,float);//为对象置姓名、成绩
    void SetName(char *);   //为对象置姓名
    char *GetName(void);    //取得学生姓名
};

float Stu::Average(void){  return (Chinese+Math)/2;}//平均成绩
float Stu::Sum(void){    return Chinese+Math;  }//总分
void Stu::Show(void)    //打印信息
{   cout<<"Name:  "<<Name<<endl<<"Score:  "<<Chinese<<'\t'<<
   Math<<'\t'<<"average:  "<<Average()<<'\t'<<"Sum:   "<<Sum()<<endl;
  }
void Stu::SetStudent(char *name,float chinese,float math)
{   strcpy(Name,name);  //置姓名
    Chinese=chinese;    //置语文成绩
    Math=math;      //置数学成绩
}
char * Stu::GetName(void){  return Name;}//返回姓名


int main()
{
    Extended_queue<int> q;
    q.append(1);
    q.append(2);
    q.append(3);
    cout << q.full() << endl;
    cout << q.size() << endl;

    int tmp = 0;
    q.serve_and_retrieve(tmp);
    cout << tmp << endl;
    tmp = 0;
    q.clear();
    q.serve_and_retrieve(tmp);
    cout << tmp << endl;
}
```

分析

delete 释放 new 分配的单个对象指针指向的内存
delete[] 释放 new 分配的对象数组指针指向的内存
delete 运算符的结果类型为 void，因此它不返回值。

运行结果截图



第二题

代码

```cpp
#include <iostream>
#include <string.h>

using namespace std;

class Stu
{
    char Name[20];  //学生姓名
    float Chinese;      //语文成绩
    float Math;     //数学成绩
public:
    Stu(char*,float,float);
    float Average(void);    //计算平均成绩
    float Sum(void);    //计算总分
    void Show(void);    //打印信息
    void SetStudent(char*,float,float);//为对象置姓名、成绩
    void SetName(char *);   //为对象置姓名
    char *GetName(void);    //取得学生姓名
};

Stu::Stu(char* x,float y,float z)
{
    strcpy(Name,x);
    Chinese = y;
    Math = z;
}

float Stu::Sum(void)
{
    return Chinese+Math;
}

char * Stu::GetName(void)
{
    return Name;
}

```

```
38  void max(Stu *t)
39  {
40      int maximum = 0;
41      Stu *tp = t;
42      for (int i=0 ; i < 5 ; i++)
43      {
44          if (t->Sum() > maximum)
45          {
46              maximum = t->Sum();
47              tp = t;
48          }
49          t++;
50      }
51      cout << tp->GetName() << endl;
52  }
53
54  int main()
55  {
56      Stu stu[5] = {Stu("name1",65,97),
57                    Stu("name2",64,35),
58                    Stu("name3",87,105),
59                    Stu("name4",90,88),
60                    Stu("name5",68,45)};
61      max(&stu[0]);
62      return 0;
63  }
```

分析：

对象数组的定义和初始化：
类类型 数组名[元素个数] = { 数组的初始化列表... };
说明：
1）在没有初始化列表的数组中，所有对象默认调用无参的构造函数。
2）对于有初始化列表的数组中，可以用构造函数来生成"类类型"的无名对象来初始化数组内的元素。

运行结果截图：

```
name3

Process returned 0 (0x0)   execution time : 0.154 s
Press any key to continue.
```

第三题

代码：

```cpp
#include <iostream>

using namespace std;

class Student
{
public:
    Student(int n,float s):num(n),score(s){}
    void change (int n ,float s){num = n;score = s;}
    void display(){cout << num << "  " << score << endl;}
private:
    int num;
    float score;
};

void fun(Student& t,int x,float y)
{
    t.change(x,y);
    t.display();
}

int main()
{
    Student stud (101,78.5);
    stud.display();
    stud.change(101,80.5);
    stud.display();
    fun(stud,54,153.5);
    return 0;
}
```

分析：

引用做参数：
形参相当于是实参的“别名”，对形参的操作其实就是对实参的操作，在引用传递过程中，被调函数的形式参数虽然也作为局部变量在栈中开辟了内存空间，但是这时存放的是由主调函数放进来的实参变量的地址。被调函数对形参的任何操作都被处理成间接寻址，即通过栈中存放的地址访问主调函数中的实参变量。正因为如此，被调函数对形参做的任何操作都影响了主调函数中的实参变量。

运行结果截图：

```
101  78.5
101  80.5
54  153.5

Process returned 0 (0x0)   execution time : 0.146 s
Press any key to continue.
```