



C++ 程序设计

C++ Programming

实验报告 3-4

Experiment Report

专业 (Major) : 计算机科学与技术 2502 班

学号 (ID) : 202512898

姓名 (Name) : 田佩宁

提交日期 (Date) : 2025 年 12 月 7 日

成绩 (Score) :



实验编号 (Experiment No.) : 3

实验名称 (Experiment Name) : 自定义数据类型的设计使用

实验目的 (Experiment Purpose) :

- (1) 掌握结构体类型变量的定义和使用;
- (2) 掌握结构体类型数组的概念和应用;
- (3) 了解链表的概念, 初步学会对简单的链表进行操作。

实验内容 (Experiment Contents) :

必做题: 设计一个结构体类型, 描述进程的结构, 如图 1 所示。然后定义一个结构体数组存储如图 2 所示的 4 个进程。初始时进程的状态都是就绪状态即 $pState=1$, 从优先级最高的进程开始执行, 将执行的进程状态 $pState$ 由就绪状态改为执行状态即 $pState=2$, 并输出当前进程信息(包括进程的每一个属性的信息), 然后该进程 $pCPU--$, 直到为 0, 将该进程状态改为 $pState=3$, 表示进程执行完毕。输出当前进程信息(包括进程的每一个属性的信息)。接下来执行剩余就绪状态进程中优先级最高的进程, 直到没有就绪状态进程为止。

选做题: 设计一个结构体类型, 描述进程的结构, 如图 1 所示。然后定义一个结构体数组或链表存储如图 2 所示的 4 个进程。初始时进程的状态都是就绪状态即 $pState=1$, 从优先级最高的进程开始执行, 将执行的进程状态 $pState$ 由就绪状态改为执行状态即 $pState=2$ 。然后不断循环, 每循环一次, 就绪状态的进程优先级增加 1, 其余不变; 执行状态的进程优先级减 3 且其 $pCPU$ 减 1; 当 $pCPU$ 为 0 时, 该进程执行完毕, 其进程状态修改为停止即 $pState=3$, 其余不再变化。直到所有进程 $pCPU$ 都为 0, 则循环结束。

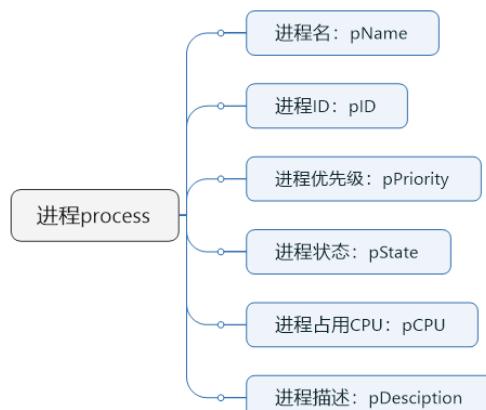


图 1. 进程的结构

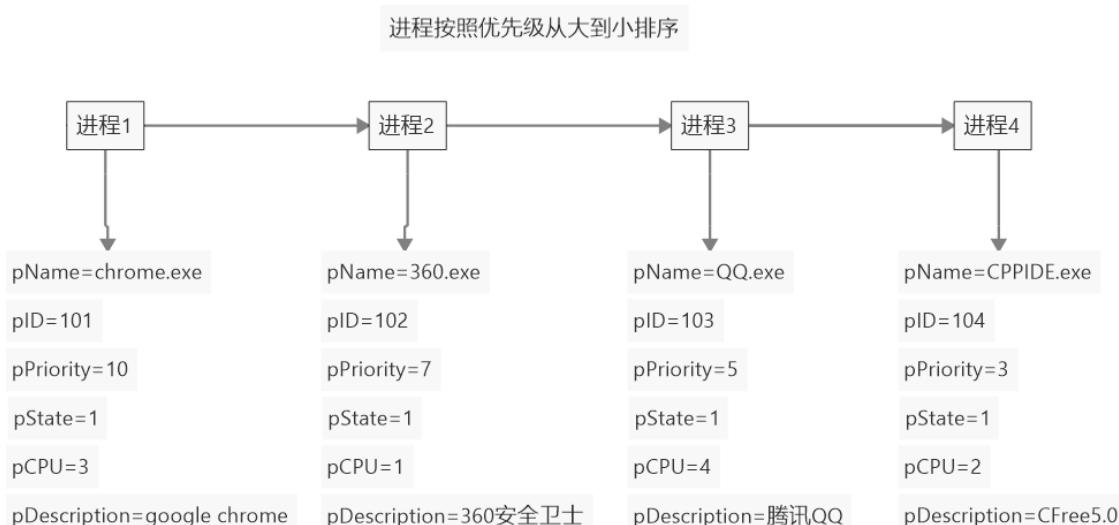


图 2. 4 个进程对象



实验结果 (Experiment Results) :

题目一

```
1 #include <algorithm>
2 #include <iostream>
3 #include <iomanip>
4 using namespace std;
5
6 struct monitor{
7     string pName, pDescription;
8     int pID, pPriority, pState, pCPU;
9     monitor(string _pName, int _pID, int _pPriority,
10            int _pState, int _pCPU, string _pDescription){
11         pName=_pName, pDescription=_pDescription;
12         pID=_pID, pPriority=_pPriority,
13         pState=_pState, pCPU=_pCPU;
14     }
15     void execute(int&cnt){
16         pState=2, cnt++;
17         while (pCPU!=0) {
18             pCPU--;
19             cout<<left<<setw(n: 2)<<cnt<<" "<<setw(n: 12)
20             <<pName<<setw(n: 5)<<pID
21             <<setw(n: 3)<<pPriority<<setw(n: 3)
22             <<pState<<setw(n: 3)<<pCPU
23             <<pDescription<<'\n';
24             cnt++;
25         }
26         cnt--, pState=3;
27         cout<<left<<setw(n: 2)<<cnt<<" "<<setw(n: 12)
28             <<pName<<setw(n: 5)<<pID
29             <<setw(n: 3)<<pPriority<<setw(n: 3)
30             <<pState<<setw(n: 3)<<pCPU<<pDescription<<'\n';
31     }
32 };
33 bool cmp(monitor a,monitor b){
34     if (a.pPriority>b.pPriority) {
35         return true;
36     }else {
37         return false;
38     }
39 }
41 //change chrome.pPriority to 1.
42 int main(){
43     monitor process[]={{[0]={"chrome.exe", 101, 1, 1, 3, "google chrome"}, 
44                         [1]={"360.exe", 102, 7, 1, 1, "360 safe protect"}, 
45                         [2]={"QQ.exe", 103, 5, 1, 4, "tencent QQ"}, //666 "tencent"
46                         [3]={"CPPIDE.exe", 104, 3, 1, 2, "CFree 5.0"},},};
47     int cnt=0;
48     sort(first: process, last: process+4, comp: cmp);
49     for (int i=0; i<4; i++) {
50         process[i].execute(&cnt);
51     }
52     cout<<"田佩宁 202512898\n";
53 }
```

1 :360.exe 102 7 2 0 360 safe protect
1 :360.exe 102 7 3 0 360 safe protect
2 :QQ.exe 103 5 2 3 tencent QQ
3 :QQ.exe 103 5 2 2 tencent QQ
4 :QQ.exe 103 5 2 1 tencent QQ
5 :QQ.exe 103 5 2 0 tencent QQ
6 :CPPIDE.exe 104 3 2 1 CFree 5.0
7 :CPPIDE.exe 104 3 2 0 CFree 5.0
7 :CPPIDE.exe 104 3 3 0 CFree 5.0
8 :chrome.exe 101 1 2 2 google chrome
9 :chrome.exe 101 1 2 1 google chrome
10:chrome.exe 101 1 2 0 google chrome
10:chrome.exe 101 1 3 0 google chrome
田佩宁 202512898



题目二

```

1 #include <iostream>
2 #include <iomanip>
3 using namespace std;
4 struct monitor{
5     string pName, pDescription;
6     int pID, pPriority, pState, pCPU;
7     monitor(string _pName, int _pID, int _pPriority,
8             int _pState, int _pCPU, string _pDescription){
9         pName=_pName, pDescription=_pDescription;
10        pID=_pID, pPriority=_pPriority,
11        pState=_pState, pCPU=_pCPU;
12    }
13    void display(int cnt){
14        cout<<left<<setw(n: 2)<<cnt<<" "<<setw(n: 12)
15            <<pName<<setw(n: 5)<<pID
16            <<setw(n: 3)<<pPriority<<setw(n: 3)
17            <<pState<<setw(n: 3)<<pCPU
18            <<pDescription<<'\n';
19    }
20};
21 int select_execute_process(monitor* obj){
22     int max_priority = 0;
23     int selected_index = -1;
24
25     for (int i=0; i<4; i++) {
26         if (obj[i].pState == 1 && obj[i].pCPU > 0) {
27             if (obj[i].pPriority > max_priority) {
28                 max_priority = obj[i].pPriority;
29                 selected_index = i;
30             }
31         }
32     }
33     return selected_index;
34 }
35
36 bool is_allzero(monitor* obj){}
37     for (int i=0; i<4; i++) {
38         if (obj[i].pCPU != 0) {
39             return false;
40         }
41     }
42     return true;
43
44 int main(){
45     monitor process[]=
46     { [0]={"chrome.exe", 101, 1, 1, 3, "google chrome"}, 
47       [1]={"360.exe", 102, 7, 1, 1, "360 safe protect"}, 
48       [2]={"QQ.exe", 103, 5, 1, 4, "tencent QQ"}, 
49       [3]={"CPPIDE.exe", 104, 3, 1, 2, "CFree 5.0"} };
50
51     int cnt=0;
52     while (is_allzero(obj: process) != true){
53         cnt++;
54         int execute_index = select_execute_process(obj: process);
55         if (execute_index != -1){
56             process[execute_index].pState = 2; //change
57         }
58
59         for (int i = 0; i < 4; i++){
60             if (process[i].pState == 1){
61                 process[i].pPriority += 1; //keep it
62             }else if (process[i].pState == 2){
63                 process[i].pPriority -= 3;
64                 process[i].pCPU -= 1; //execute
65                 process[i].display(cnt);
66                 process[i].pState = 1; //recover
67
68                 if (process[i].pCPU == 0){
69                     process[i].pState = 3;
70                 }
71                 continue;
72             }
73             process[i].display(cnt); //if no
74         }
75         cout << '\n';
76     }
77
78     cout<<"田佩宁 202512898\n";
79 }

```

1 :chrome.exe	101	2	1	3	google chrome
1 :360.exe	102	4	2	0	360 safe protect
1 :QQ.exe	103	6	1	4	tencent QQ
1 :CPPIDE.exe	104	4	1	2	CFree 5.0
2 :chrome.exe	101	3	1	3	google chrome
2 :360.exe	102	4	3	0	360 safe protect
2 :QQ.exe	103	3	2	3	tencent QQ
2 :CPPIDE.exe	104	5	1	2	CFree 5.0
3 :chrome.exe	101	4	1	3	google chrome
3 :360.exe	102	4	3	0	360 safe protect
3 :QQ.exe	103	4	1	3	tencent QQ
3 :CPPIDE.exe	104	2	2	1	CFree 5.0
4 :chrome.exe	101	1	2	2	google chrome
4 :360.exe	102	4	3	0	360 safe protect
4 :QQ.exe	103	5	1	3	tencent QQ
4 :CPPIDE.exe	104	3	1	1	CFree 5.0
5 :chrome.exe	101	2	1	2	google chrome
5 :360.exe	102	4	3	0	360 safe protect
5 :QQ.exe	103	2	2	2	tencent QQ
5 :CPPIDE.exe	104	4	1	1	CFree 5.0
6 :chrome.exe	101	3	1	2	google chrome
6 :360.exe	102	4	3	0	360 safe protect
6 :QQ.exe	103	3	1	2	tencent QQ
6 :CPPIDE.exe	104	1	2	0	CFree 5.0
7 :chrome.exe	101	0	2	1	google chrome
7 :360.exe	102	4	3	0	360 safe protect
7 :QQ.exe	103	4	1	2	tencent QQ
7 :CPPIDE.exe	104	1	3	0	CFree 5.0
8 :chrome.exe	101	1	1	1	google chrome
8 :360.exe	102	4	3	0	360 safe protect
8 :QQ.exe	103	1	2	1	tencent QQ
8 :CPPIDE.exe	104	1	3	0	CFree 5.0
9 :chrome.exe	101	-2	2	0	google chrome
9 :360.exe	102	4	3	0	360 safe protect
9 :QQ.exe	103	2	1	1	tencent QQ
9 :CPPIDE.exe	104	1	3	0	CFree 5.0
10:chrome.exe	101	-2	3	0	google chrome
10:360.exe	102	4	3	0	360 safe protect
10:QQ.exe	103	-1	2	0	tencent QQ
10:CPPIDE.exe	104	1	3	0	CFree 5.0

田佩宁 202512898



心得体会 (Experience) :

今日 C++ 实验，亲手定制数据类型，悟封装之魅，享调试之趣，收获满满。

实验编号 (Experiment No.) : 4

实验名称 (Experiment Name) : 继承与多重继承

实验目的 (Experiment Purpose) :

掌握类的定义方法、对象和构造函数的使用；掌握并灵活使用 C++ 中类的继承与派生的基本用法，掌握多重继承的思想和用法；初步了解继承与派生方法在模块化程序设计中的作用。

实验内容 (Experiment Contents) :

(1) 创建一个学生类，成员变量有姓名，学号和三门课 C++ 和高等数学和大学英语的成绩，成员函数有求每个学生最高分和平均分的函数：max() 和 avg()。如图 1 所示。初始化 5 个学生的信息，然后按照平均分从大到小排序后输出 5 个学生信息（包括姓名、学号、平均分），再按照最高分从大到小排序后输出 5 个学生信息（包括姓名、学号、最高成绩和对应课程名）。

要求：创建一个工程（project），将 Student 类的声明放到头文件 student.h 中，将 Student 类的成员函数的定义放到源文件 student.cpp 中，将 5 个学生对象的初始化以及最高成绩和平均成绩的计算输出写到源文件 main.cpp 中。如图 2（codeblocks）和图 3（CFree）所示。从而学习多文件程序的编写。

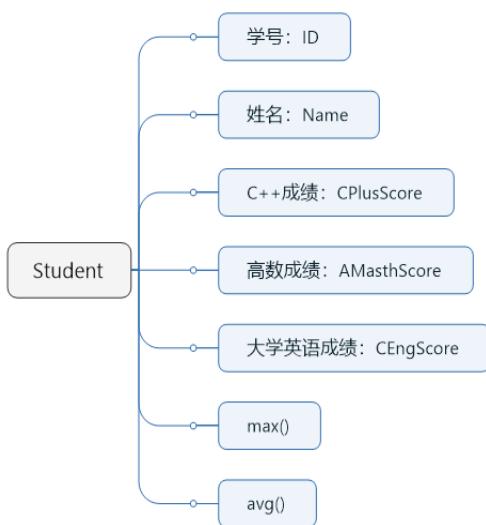


图 1

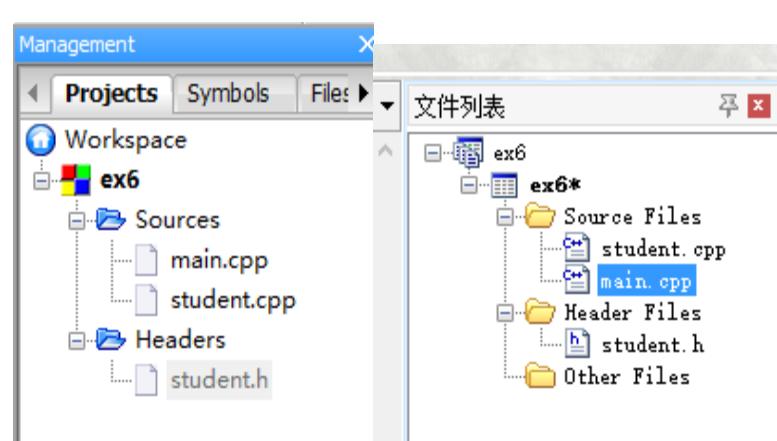


图 2

图 3

(2) 分别声明 Teacher (教师) 类和 Cadre (干部) 类，采用多重继承方式由这两个类派生出新类 Teacher_Cadre (教师兼干部) 类。要求：

- 1) 在两个基类中都包含姓名、年龄、性别、地址、电话等数据成员；
- 2) 在 Teacher 类中还包含数据成员 title (职称)，在 Cadre 类中还包含数据成员 post (职务)，在 Teacher_Cadre 类中还包含数据成员 wages (工资)；
- 3) 对两个基类中的姓名、年龄、性别、地址、电话等数据成员用相同的名字，在引用这些数据成员时，指定作用域；



4) 在类体中声明成员函数，在类外定义成员函数；

5) 在派生类 Teacher_Cadre 的成员函数 show 中调用 Teacher 类中的 display 函数，输出姓名、年龄、性别、职称、地址、电话，然后再用 cout 语句输出职务与工资。

实验结果 (Experiment Results) :

题目一

```
1 #ifndef __Class_Student
2 #define __Class_Student
3
4 #include <iostream>
5 class Student{
6     std::string Name;
7     int ID, CPlusScore, AMathScore, CEngScore;
8 public:
9     Student(){};
10    void init();
11    int max();
12    double avg();
13 };
14
15 #endif

1 #include <iostream>
2 #include "Student.h"
3 using namespace std;
4
5 void Student::init(){
6     int ID;      cin>>ID;      this->ID = ID;
7     string Name;  cin>>Name;   this->Name = Name;
8     int CPlusScore;  cin>>CPlusScore;this->CPlusScore = CPlusScore;
9     int AMathScore;  cin>>AMathScore;this->AMathScore = AMathScore;
10    int CEngScore;  cin>>CEngScore; this->CEngScore = CEngScore;
11 }
12 int Student::max(){
13     int temp[] = {[0]=this->CPlusScore, [1]=this->AMathScore, [2]=this->CEngScore,};
14     int max = 0;
15     for (int i=0; i<3; i++) {if (temp[i]>max) {max=temp[i];}}
16     return max;
17 }
18 double Student::avg(){
19     double avg = (this->CPlusScore + this->AMathScore + this->CEngScore) / 3.0;
20     return avg;
21 }

1 #include <iostream>
2 #include "Student.h"
3 #include <iomanip>
4 using namespace std;
5 const int STUN = 5;
6
7 double MaxAvg(Student *p){
8     int sum = 0;
9     for (int i=0; i<STUN; i++) {
10         sum += p[i].max();
11     }
12     return static_cast<double>(sum) / STUN;
13 }
14 double GlbAvg(Student *p){
15     int sum = 0;
16     for (int i=0; i<STUN; i++) {
17         sum += p[i].avg();
18     }
19     return static_cast<double>(sum) / STUN;
20 }

21 int main(){
22     Student a[STUN];
23     for (int i=0; i<STUN; i++) {
24         a[i].init();
25     }
26     cout<<fixed<<setprecision(n:1)<<MaxAvg(p: a)<<"\n";
27     <<fixed<<setprecision(n:1)<<GlbAvg(p: a)<<"\n";
28     90.0
29     cout<<"田佩宁 202512898\n";
30
31 }
```

1001 Kim 99 95 97
1002 Sam 89 78 95
1003 Tim 90 95 97
1004 Kathy 60 53 70
1005 Jane 70 89 63
90.0
82.6 田佩宁 202512898



题目二

```
1 #include <iostream>
2 #include <string>
3 using namespace std;
4
5 // ***declaration***
6 class Teacher {
7 protected:
8     string name, age, sex, address, phone;
9     string title;
10 public:
11     Teacher(string n, string a, string s,
12             |   string t, string addr, string tel);
13     void display() const;
14 };
15
16 class Cadre {
17 protected:
18     string name, age, sex, address, phone;
19     string post;
20 public:
21     Cadre(string n, string a, string s,
22           |   string p, string addr, string tel);
23 };
24
25 class Teacher_Cadre : public Teacher, public Cadre {
26 protected:
27     double wages;
28 public:
29     Teacher_Cadre(string n, string a, string s,
30                 |   string t, string p, string addr,
31                 |   string tel, double w);
32     void show() const;
33 };
34
35 // ***defination***
36 // class Teacher
37 Teacher::Teacher(string n, string a, string s,
38                  string t, string addr, string tel)
39     : name(std::move(n)), age(std::move(a)),
40       sex(std::move(s)), title(std::move(t)),
41       address(std::move(addr)), phone(std::move(tel)) {}
42
43 void Teacher::display() const {
44     cout << name << ' ' << age << ' ' << sex << ' '
45     |   | << title << ' ' << address << ' ' << phone;
46 }
47 // class Cadre
48 Cadre::Cadre(string n, string a, string s,
49               string p, string addr, string tel)
50     : name(std::move(n)), age(std::move(a)),
51       sex(std::move(s)), post(std::move(p)),
52       address(std::move(addr)), phone(std::move(tel)) {}
53
54 // class Teacher_Cadre
55 Teacher_Cadre::Teacher_Cadre(string n, string a, string s,
56                             string t, string p, string addr, string tel, double w)
57     : Teacher(n, a, s, t, addr, tel),
58       Cadre(n, a, s, p, addr, tel), wages(w) {}
58
59 void Teacher_Cadre::show() const {
60     Teacher::display();
61     cout << ' ' << Cadre::post << ' ' << wages << endl;
62 }
63
64
65 int main() {
66     string name, age, sex, title, addr, phone, post;
67     double wages;
68     cin >> name >> age >> sex >> title
69     |   | >> addr >> phone >> post >> wages;
70
71     Teacher_Cadre ng(n: name, a: age, s: sex,
72                       t: title, p: post, addr: phone, w: wages);
73     ng.show();
74
75     cout << "田佩宁 202512898\n";
76 }
```



```
workspaces/Penacony/lecture/"EXP4_2 && rm EXP4_2  
ZhangSan 18 male lecture TaishanRoad143 18633335555 principal 90000  
ZhangSan 18 male lecture TaishanRoad143 18633335555 principal 90000  
田佩宁 202512898
```

心得体会 (Experience) :

多继承逻辑复杂，虚基类解菱形冲突，实践深化理解，代码更需严谨设计。