

河南工业大学

《面向对象程序设计》实验报告

专业班级： 软件 1305 班 学号： 201316920311 姓名： 田劲锋

实验单元一 类和对象

实验一 标准控制台输入输出

实验时间： 2014 年 11 月 28 日

【实验目的】

- 1、 熟悉 Dev-Cpp 编程环境。
- 2、 编写简单的输入输出语句。
- 3、 熟练使用算术运算符。
- 4、 能够编写简单的判断语句。

【实验环境】

GNU C++ version 3.4.5 (mingw-vista special r3)

【实验内容】

编写 C++ 程序，实现输入两个整数，输出两个整数的加、减、乘、除结果；详细的注释，完整的输出显示。

【详细分析】

实验内容是对两个输入整数的四则运算，即简单的顺序结构。

在此基础上稍作改动，即输入整数和运算符来自动进行计算，流程如图1。

【实验源码】

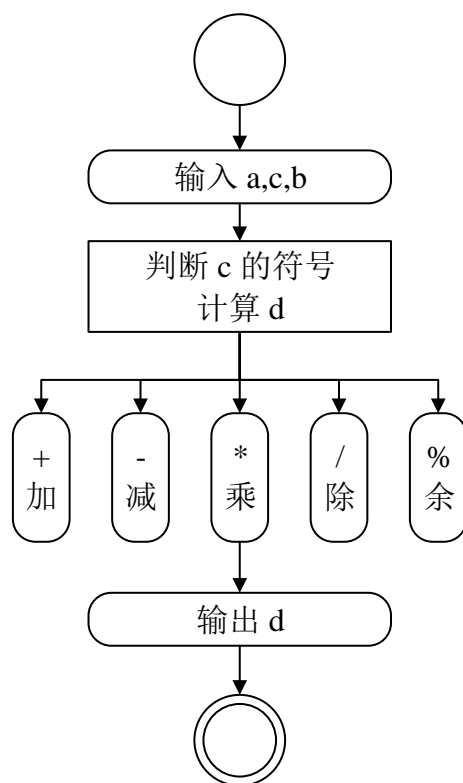


图 1: 程序流程图

Listing 1: exp01.cpp

```

1  #include <iostream>
2  /* 输入输出流 */
3  using namespace std;
4  /* std 命名空间 */
5  int main()
6  /* 主函数 */
7  {
8      int a, b;
9      /* 声明两个整数 */
10     char c;
11     /* 声明一个字符 */
12     cin >> a >> c >> b;
13     /* 从屏幕输入流读入 */
14     int d = 0;
15     /* 初始化结果 */
16     switch(c) {
17     /* 分支判断运算符 */
18         case '+':
19             d = a + b;
20             break;
21             /* break 是 C 遗留的冗余语句 */
22         case '-':
23             d = a - b;
24             break;
25         case '*':
26             d = a * b;
27             break;
28         case '/':
29             d = a / b;
30             break;
31         case '%':
32             d = a % b;
33             break;
34         /* 不需要 default */
35     }
36     cout << d << endl;
37     /* 输出结果并换行 */
38     return 0;
39     /* 正常结束返回 0 */
40 }

```

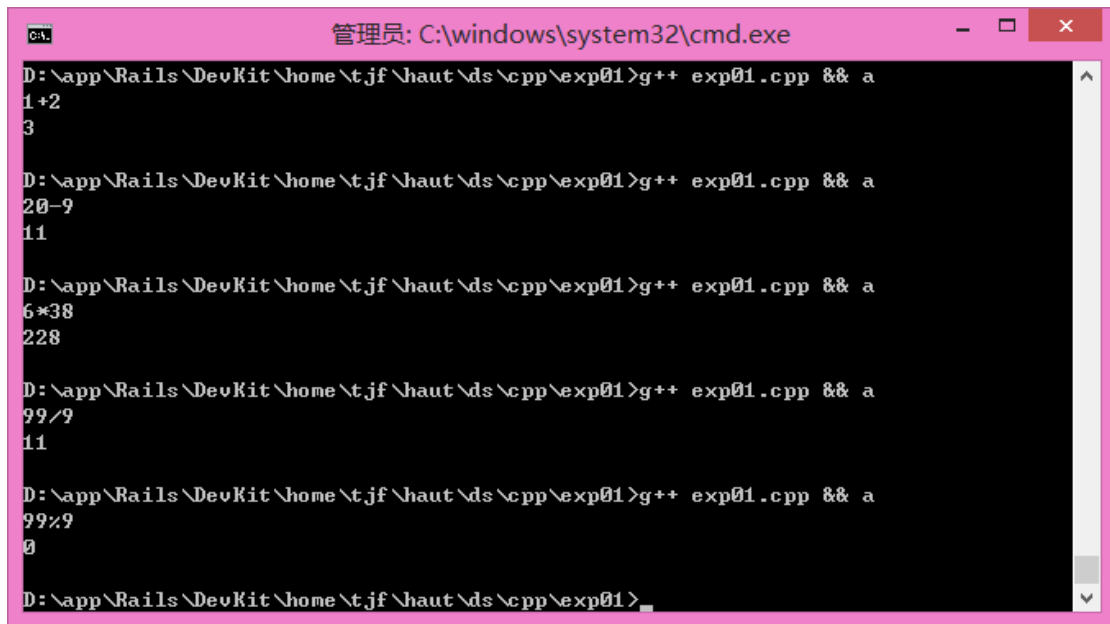
【实验结果】

图2显示了编译、运行、输入、输出的过程。

【实验体会】

这是一个非常基础的简单程序，目的在于熟悉编程和调试环境。程序本身没有任何难度，加上注释也不过40行。

关于编程环境的配置，我倾向于使用编辑器（如 Vim、Sublime Text）编写



```
管理员: C:\windows\system32\cmd.exe
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp01>g++ exp01.cpp && a
1+2
3
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp01>g++ exp01.cpp && a
20-9
11
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp01>g++ exp01.cpp && a
6*38
228
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp01>g++ exp01.cpp && a
99/9
11
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp01>g++ exp01.cpp && a
99%9
0
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp01>
```

图 2: 标准控制台输入输出

源文件，在命令行下编译、运行和调试（使用 gcc/gdb）。当然，Microsoft Visual Studio 作为世界上最好的 IDE，在编写调试程序中也是非常好用的，所以在有条件的时候也会使用 VS。

实验二 类和对象

实验时间： 2014 年 11 月 29 日

【实验目的】

- 1、 掌握类、对象、数据成员、成员函数的基本概念。
- 2、 能够进行类的定义。
- 3、 能够使用成员函数进行相关调用。

【实验环境】

GNU C++ version 3.4.5 (mingw-vista special r3)

【实验内容】

- 1、 编写NumberA类，实现两个整数的加减乘除运算。构造函数实现两整数a，b赋值。
- 2、 编写OperaN类，实现输入1.2.3.4解析成加减乘除符号。
- 3、 P89: 3.11

2.1 NumberA 类

【详细分析】

NumberA 类设有两个成员变量存放两个操作数，提供对这两个操作数进行四则运算的方法。

【实验源码】

Listing 2: exp01.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  class NumberA
5  {
6  private:
7      int a, b;
8
9  public:
10     NumberA(int _a, int _b) : a(_a), b(_b) {}
```

```

11     int plus() { return a + b; }
12     int minus() { return a - b; }
13     int times() { return a * b; }
14     int divide() { return a / b; }
15     int mod() { return a % b; }
16 };
17
18 int main()
19 {
20     int a, b;
21     cin >> a >> b;
22     NumberA t(a, b);
23     cout << t.plus() << endl;
24     cout << t.minus() << endl;
25     cout << t.times() << endl;
26     cout << t.divide() << endl;
27     cout << t.mod() << endl;
28     return 0;
29 }

```

【实验结果】

```

管理员: C:\windows\system32\cmd.exe
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp01.cpp && a
9 4
13
5
36
2
1

D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp01.cpp && a
12 7
19
5
84
1
5

D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>

```

图 3: NumberA 类

2.2 OperaN 类

【详细分析】

用一个整数初始化类，置符号。

【实验源码】

Listing 3: exp02.cpp

```

1 #include <iostream>

```

```

2  #include <string>
3  using namespace std;
4
5  class OperaN
6  {
7  private:
8      int num;
9
10 public:
11     char mark;
12     OperaN(int _x = 1) : num(_x) { getMark(); }
13     char getMark()
14     {
15         static string marks = "+-*/";
16         return mark = marks[num % 4];
17     }
18 };
19
20 int main()
21 {
22     int n;
23     cin >> n;
24     OperaN t(n);
25     cout << t.getMark() << endl;
26     cout << t.mark << endl;
27     return 0;
28 }

```

【实验结果】

```

管理员: C:\windows\system32\cmd.exe
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
2
*
*
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
3
/
/
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
4
+
+
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
1
-
-
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
5
-
-
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>

```

图 4: OperaN 类

2.3 P89: 3.11

【详细分析】

对 Account 类的修改。

【实验源码】

Listing 4: GradeBook.h

```
1  /* GradeBook.h */
2  #include <string>
3  using std::string;
4
5  class GradeBook
6  {
7  public:
8      GradeBook(string, string);
9      void setCourseName(string);
10     string getCourseName();
11     void setTeacherName(string);
12     string getTeacherName();
13     void displayMessage();
14
15 private:
16     string courseName;
17     string teacherName;
18 };
```

Listing 5: GradeBook.cpp

```
1  /* GradeBook.cpp */
2  #include <iostream>
3  using std::cout;
4  using std::endl;
5
6  #include "GradeBook.h"
7
8  GradeBook::GradeBook(string name, string teacher)
9  {
10     setCourseName(name);
11     setTeacherName(teacher);
12 }
13
14 void GradeBook::setCourseName(string name)
15 {
16     if (name.length() <= 25)
17         courseName = name;
18     if (name.length() > 25) {
19         courseName = name.substr(0, 25);
20         cout << "名称\" << name << "\" 长度超限 (25) 。\n"
21             << "截取前 25 个字符。\" << endl;
22     }
23 }
```



```

24
25 string GradeBook::getCourseName()
26 {
27     return courseName;
28 }
29
30 void GradeBook::setTeacherName(string name)
31 {
32     if (name.length() <= 25)
33         teacherName = name;
34     if (name.length() > 25) {
35         teacherName = name.substr(0, 25);
36         cout << "姓名\" << name << "\" 长度超限 (25) 。\n"
37             << "截取前 25 个字符。\\n" << endl;
38     }
39 }
40
41 string GradeBook::getTeacherName()
42 {
43     return teacherName;
44 }
45
46 void GradeBook::displayMessage()
47 {
48     cout << "欢迎使用 " << getCourseName() << " 课程表! \n"
49         << "任课教师: " << getTeacherName() << endl;
50 }

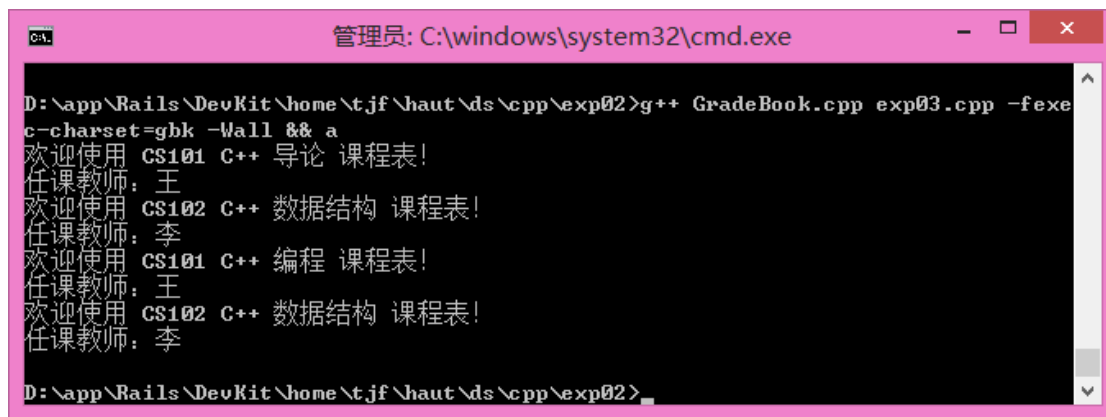
```

Listing 6: 主程序 exp03.cpp

```

1  #include <iostream>
2  using std::cout;
3  using std::endl;
4
5  #include "GradeBook.h"
6
7  int main()
8  {
9      GradeBook gradeBook1("CS101 C++ 导论", "王");
10     GradeBook gradeBook2("CS102 C++ 数据结构", "李");
11
12     gradeBook1.displayMessage();
13     gradeBook2.displayMessage();
14
15     gradeBook1.setCourseName("CS101 C++ 编程");
16
17     gradeBook1.displayMessage();
18     gradeBook2.displayMessage();
19     return 0;
20 }

```



```
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ GradeBook.cpp exp03.cpp -fexe
c-charset=gbk -Wall && a
欢迎使用 CS101 C++ 导论 课程表!
任课教师: 王
欢迎使用 CS102 C++ 数据结构 课程表!
任课教师: 李
欢迎使用 CS101 C++ 编程 课程表!
任课教师: 王
欢迎使用 CS102 C++ 数据结构 课程表!
任课教师: 李
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>
```

图 5: P89: 3.11

【实验结果】

【实验体会】

(至少150字)

实验三 结构化控制结构

实验时间： 2014 年 11 月 30 日

【实验目的】

- 1、 掌握基本的结构化控制结构。
- 2、 能够熟练进行结构化编程。

【实验环境】

GNU C++ version 3.4.5 (mingw-vista special r3)

【实验内容】

- 1、 编写NumberA类，实现两个整数的加减乘除运算，可以循环计算。构造函数实现两整数a，b赋值。
- 2、 P177： 5.29

3.1 NumberA 类

【详细分析】

NumberA 类设有两个成员变量存放两个操作数，提供对这两个操作数进行四则运算的方法。

主程序循环读入两个整数，进行运算并输出。

【实验源码】

Listing 7: exp01.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  class NumberA
5  {
6  private:
7      int a, b;
8
9  public:
10     NumberA(int _a, int _b) : a(_a), b(_b) {}
11     int plus() { return a + b; }
12     int minus() { return a - b; }
```

```

13     int times() { return a * b; }
14     int divide() { return a / b; }
15     int mod() { return a % b; }
16 };
17
18 int main()
19 {
20     int a, b;
21     while (cin >> a >> b) {
22         NumberA t(a, b);
23         cout << a << '+' << b << '=' << t.plus() << endl;
24         cout << a << '-' << b << '=' << t.minus() << endl;
25         cout << a << '*' << b << '=' << t.times() << endl;
26         cout << a << '/' << b << '=' << t.divide() << endl;
27         cout << a << '%' << b << '=' << t.mod() << endl;
28     }
29     return 0;
30 }

```

【实验结果】

```

管理员: C:\windows\system32\cmd.exe
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp03>g++ exp01.cpp -Wall && a
1 2
1+2=3
1-2=-1
1*2=2
1/2=0
1%2=1
5 3
5+3=8
5-3=2
5*3=15
5/3=1
5%3=2
9 14
9+14=23
9-14=-5
9*14=126
9/14=0
9%14=9
^Z
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp03>

```

图 6: NumberA 类

3.2 OperaN 类

【详细分析】

用一个整数初始化类，置符号。

【实验源码】

Listing 8: exp02.cpp

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class OperaN
6  {
7  private:
8      int num;
9
10 public:
11     char mark;
12     OperaN(int _x = 1) : num(_x) { getMark(); }
13     char getMark()
14     {
15         static string marks = "+-*/";
16         return mark = marks[num % 4];
17     }
18 };
19
20 int main()
21 {
22     int n;
23     cin >> n;
24     OperaN t(n);
25     cout << t.getMark() << endl;
26     cout << t.mark << endl;
27     return 0;
28 }
```

【实验结果】

【实验体会】

(至少150字)

```
管理员: C:\windows\system32\cmd.exe
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
2
*
*
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
3
/
/
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
4
+
+
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
1
-
-
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>g++ exp02.cpp -Wall && a
5
-
-
D:\app\Rails\DevKit\home\tjf\haut\ds\cpp\exp02>
```

图 7: OperaN 类