

云南大学数学与统计学实验教学中心

《高级语言程序设计》实验报告

课程名称：程序设计和算法语言	学期：2016~2017 学年上学期	成绩：
指导教师：赵越	学生姓名：刘鹏	学生学号：20151910042
实验名称：选择结构程序设计		
实验编号：No.03	实验日期：2018 年 8 月 9 日	实验学时：2
学院：数学与统计学院	专业：信息与计算科学	年级：2015 级

一、实验目的

1. 熟练掌握上机运行一个 C 程序的操作过程。
2. 学会正确使用逻辑运算符和逻辑表达式，进一步掌握各种表达式的使用。
3. 利用 if 语句实现选择结构。
4. 利用 switch 语句实现多分支选择结构。
5. 熟悉关系表达式和逻辑表达式的使用。
6. 掌握 break 和 continue 语句的使用，以及它们之间的区别。
7. 进一步练习调试与修改程序。

二、实验环境

Windows10 Pro Workstation 17096;
Code::Blocks 16.01 GCC 集成开发环境;
Cygwin GCC 编译器。

三、实验内容

1.1 1 题

三个整数 a , b , c ，由键盘输入这三个数,求三个数中最大的值。运行下面程序，分析 if 和 else 是哪两个相互“配对”，在书写程序时，分出层次，这样有利于程序的可读性，容易查找出错误。

1.1.1 程序代码

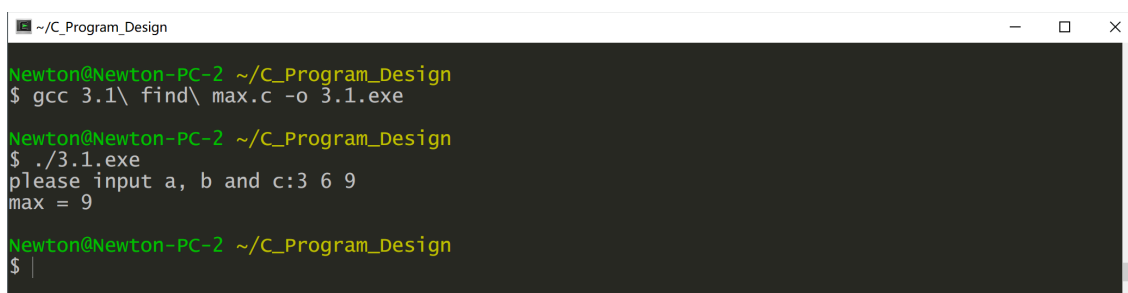
```
1  /*
2  * filename: 3.1 find max.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int a,b,c;
```

```

10  printf("please input a, b and c:");
11  scanf("%d %d %d", &a, &b, &c);
12  if(a < b) {
13      if(b < c) {
14          printf("max = %d\n", c);
15      }
16      else {
17          printf("max = %d\n", b);
18      }
19  }
20  else {
21      if(a < c) {
22          printf("max = %d\n", c);
23      }
24      else {
25          printf("max = %d\n", a);
26      }
27  }
28  return 0;
29  }

```

1.1.2 运行结果



```

~/C_Program_Design
Newton@Newton-PC-2 ~/_C_Program_Design
$ gcc 3.1\ find\ max.c -o 3.1.exe
Newton@Newton-PC-2 ~/_C_Program_Design
$ ./3.1.exe
please input a, b and c:3 6 9
max = 9
Newton@Newton-PC-2 ~/_C_Program_Design
$

```

此程序还有更加简明的方法实现，就是利用条件表达式。

```

1  #include <stdio.h>
2
3  int main() {
4      int a, b, c, max, t;
5      printf("input a, b, c:");
6      scanf("%d %d %d", &a, &b, &c);
7      t = (a > b)? a : b;
8      max = (t > c)? t : c;
9      printf("max = %d", max);
10     return 0;
11 }

```

1.2 2 题

先读下面程序，分析出程序的执行结果，然后再上机运行，结果是否一致。

1.2.1 程序代码

```
1  /*
2  * filename: 3.2 example.c
3  * property: bad example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int x, y = 1;
10     int z;
11     if(y != 0) {
12         x = 5;
13         printf("x = %d\t", x);
14     }
15     if(y == 0) {
16         x = 3;
17     }
18     else {
19         x = 5;
20         printf("x = %d\t\n", x);
21     }
22     z = 1;
23     if(z < 0) {
24         if(y > 0) {
25             x = 3;
26         }
27         else {
28             x = 5;
29         }
30         printf("x = %d\t\n", x);
31     }
32     if(1 == (z = (y < 0))) {
33         x = 3;
34     }
35     else {
36         if(y == 0) {
37             x = 5;
38         }
39         else {
40             x = 7;
41         }
42         printf("x = %d\t", x);
43         printf("z = %d\t\n", z);
44     }
45     if(1 == (x = (z = y))) {
46         x = 3;
47     }
```

```

48     printf("x = %d\t", x);
49     printf("z = %d\t\n", z);
50     return 0;
51 }

```

1.2.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.2\ example.c -o 3.2.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.2.exe
x = 5   x = 5
x = 7   z = 0
x = 3   z = 1

Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.3 题 3

有一函数

$$y = \begin{cases} x, & x < 0 \\ 3x - 2, & 10 \leq x < 50 \\ 4x + 1, & 50 \leq x < 100 \\ 5x, & x \geq 100 \end{cases}$$

输入 x 的值，求 y 的值。

1.3.1 程序代码

```

1  /*
2  * filename: 3.3 case example.c
3  * property: example
4  */
5
6  #include<stdio.h>
7
8  int main() {
9      int x, y, t;
10     printf("please input the value of x:\n");
11     scanf("%d", &x);
12     if(x < 10) {
13         t = 0;
14     }
15     if(x >= 100) {
16         t = 10;
17     }
18     else {
19         t = x / 10;
20     }
21     switch(t) {
22         case 0: y = x; break;

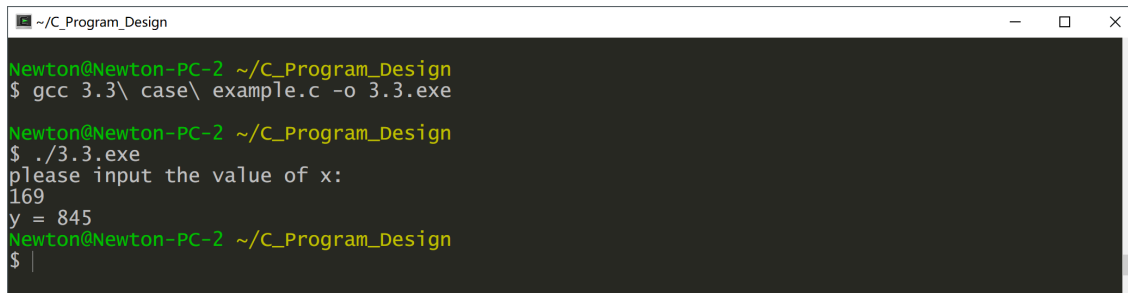
```

```

23     case 4: y = 3 * x - 2; break;
24     case 9: y = 4 * x + 1; break;
25     case 10: y = 5 * x;
26 }
27 printf("y = %d", y);
28 return 0;
29 }

```

1.3.2 运行结果



```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.3\ case\ example.c -o 3.3.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.3.exe
please input the value of x:
169
y = 845
Newton@Newton-PC-2 ~/C_Program_Design
$

```

本题还可以单独用 if 语句实现，方法简单，程序可读性好，学生自己编程，上机运行。

1.4 题 4

从数字 1 开始到 200 之间，求能被 3 整除的数；然后求这些数的累加和，直到和的值不大于 100 为止。输出这些数及累加和。

1.4.1 程序代码

```

1  /*
2  * filename: 3.4 conditional sum.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int i, t, sum = 3;
10     for(i = 1; i <= 200; i++) {
11         if(i % 3 != 0) {
12             continue;
13         }
14         printf("i = %-4d", i);
15         if(i % 5 == 0) {
16             printf("\n");
17         }
18     }
19     printf("\n");
20     for(t = 3; t <= 200; t += 3) {
21         sum = sum + t;
22         printf("t = %d\n", t);

```

```

23     if(sum > 100) {
24         break;
25     }
26 }
27 sum = sum - t;
28 printf("sum = %d,", sum);
29 return 0;
30 }

```

1.4.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.4\ conditional\ sum.c -o 3.4.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.4.exe
i = 3 i = 6 i = 9 i = 12 i = 15
i = 18 i = 21 i = 24 i = 27 i = 30
i = 33 i = 36 i = 39 i = 42 i = 45
i = 48 i = 51 i = 54 i = 57 i = 60
i = 63 i = 66 i = 69 i = 72 i = 75
i = 78 i = 81 i = 84 i = 87 i = 90
i = 93 i = 96 i = 99 i = 102 i = 105
i = 108 i = 111 i = 114 i = 117 i = 120
i = 123 i = 126 i = 129 i = 132 i = 135
i = 138 i = 141 i = 144 i = 147 i = 150
i = 153 i = 156 i = 159 i = 162 i = 165
i = 168 i = 171 i = 174 i = 177 i = 180
i = 183 i = 186 i = 189 i = 192 i = 195
i = 198
t = 3
t = 6
t = 9
t = 12
t = 15
t = 18
t = 21
t = 24
sum = 87,
Newton@Newton-PC-2 ~/C_Program_Design
$

```

此程序的目的在于理解 `continue` 语句的用法。学生可以自己选做一些题目理解 `break` 和 `continue` 语句之间的区别，以免混淆。

注：`break` 语句可以从循环体内跳出循环体外，提前结束循环，接着循环着下面的语句（从第三题可看出）。`continue` 语句是结束本次循环，即跳过循环体中下面尚未招待的语句，接着进行下一次执行循环的判定，即加速循环。

1.5 题 5（选做）

学生自己编程，上机调试，并记录下运行的结果。由键盘输入三个数，计算以这三个数为边长的三角形面积。提示：编程时要考虑到能构成三角形的条件为：两边之和大于第三边。求三角形面积公式为海伦公式。

1.5.1 程序代码

```

1  /*
2  * filename: 3.5 triangle area.c
3  * property: example
4  */
5
6  #include <stdio.h>

```

```

7  #include <math.h>
8
9  int main() {
10     float a, b, c, s, s1;
11     printf("please enter 3 reals:\n");
12     scanf("%f %f %f",&a, &b, &c);
13     if((a + b) > c && (a + c) > b && ( b + c ) > a) {
14         s = (a + b + c) * 0.5;
15         s1 = s * (s - a) * (s - b)*(s - c);
16         s = sqrt(s1);
17         printf("area of the triangle is %4.2f\n", s);
18     }
19     else {
20         printf("It is not triangle!\n");
21     }
22     return 0;
23 }

```

1.5.2 运行结果



```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.5\ triangle\ area.c -o 3.5.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.5.exe
please enter 3 reals:
6 6 6
area of the triangle is 15.59

Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.6 自增运算符实验

1.6.1 程序代码

```

1  /*
2   * filename: 3.6 increment operator.c
3   * property: example
4   */
5
6  #include <stdio.h>
7
8  int main() {
9     int i, j, n, m;
10     i = 8;
11     j = 10;
12     m = ++i;
13     n = j++;
14
15     printf("%d, %d, %d, %d", i, j, m, n);
16     return 0;

```

17 }

1.6.2 运行结果

```
~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.6\ increment\ operator.c -o 3.6.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.6.exe
9, 11, 9, 10
Newton@Newton-PC-2 ~/C_Program_Design
$
```

分别作以下改变并运行：

1.6.3 修改 1

将程序第 12~13 行改为：`m=i++;` `n=++j;`

1.6.3.1 运行结果

```
~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.7\ changed\ example.c -o 3.7.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.7.exe
9, 11, 8, 11
Newton@Newton-PC-2 ~/C_Program_Design
$
```

1.6.4 修改 2

程序改为如下形式，再次运行。

1.6.4.1 程序代码

```
1  /*
2  * filename: 3.8 changed example.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int i, j, n, m;
10     i = 8;
11     j = 10;
12
13     printf("%d, %d", i++, j++);
14     return 0;
15 }
```


1.6.4.2 运行结果

```
~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.8\ changed\ example.c -o 3.8.exe
Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.8.exe
8, 10
Newton@Newton-PC-2 ~/C_Program_Design
$
```

1.6.5 修改 3

1.6.5.1 程序代码

```
1  /*
2  * filename: 3.9 changed example.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int i, j, n, m;
10     i = 8;
11     j = 10;
12
13     m += i++;
14     n -= --j;
15
16     printf("%d, %d, %d, %d", i, j, m, n);
17     return 0;
18 }
```

1.6.5.2 运行结果

```
~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.9\ changed\ example.c -o 3.9.exe
Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.9.exe
9, 9, 8, -9
Newton@Newton-PC-2 ~/C_Program_Design
$
```

1.7 计算分段函数

计算下列分段函数值：

$$f(x) = \begin{cases} x^2 + x - 6, & x < 0 \text{ and } x > 3 \\ x^2 - 5x + 6, & 0 \leq x < 10 \text{ and } x \neq 2 \text{ and } x \neq 3 \\ x^2 - x - 1, & \text{else} \end{cases}$$

具体要求如下：

- ①用 **if** 语句实现分支。自变量 x 与函数值均用单精度类型。
- ②自变量 x 用 **scanf** 函数输入，且输入前要有提示。结果的输出采用以下形式：
 x =具体值， $f(x)$ =具体值
- ③分别输入 $x = -0.5, -3.0, 1.0, 2.0, 2.5, 3.0, 5.0$ ，运行该程序。

1.7.1 程序代码

```

1  /*
2  * filename: 3.10 sectioned function.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      float x;
10     float ans;
11
12     printf("Please input x: ");
13     scanf("%f", &x);
14
15     if (x < 0 && x > 3) {
16         ans = x * x + x - 6;
17     }
18     else {
19         if (x >= 0 && x < 10 && x != 2 && x != 3) {
20             ans = x * x - 5 * x + 6;
21         }
22         else {
23             ans = x * x - x - 1;
24         }
25     }
26
27     printf("f(x) = %3.2f", ans);
28 }
```

1.7.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.10\ sectioned\ function.c -o 3.10.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.10.exe
Please input x: 2
f(x) = 1.00

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.10.exe
Please input x: 666
f(x) = 442889.00

Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.8 程序分析

3. 先静态分析以下程序的运行结果，然后上机验证。

```

1  /*
2  * filename: 3.11 binary operator.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int a = 3, b = 4, c = 5, x, y, z;
10     x = c > b > a;
11     y = !a + b < c && (b != c);
12     z = c / b + ((float)a / b && (float)(a / c));
13     printf("x = %d, y = %d, z = %d\n", x, y, z);
14     x = a || b--;
15     y = a-- -3 && b;
16     printf("%d, %d, %d, %d, %d, %d", a, b, c, x, y, z);
17     return 0;
18 }

```

1.8.1 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.11\ binary\ operator.c -o 3.11.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.11.exe
x = 0, y = 1, z = 1
2, 4, 5, 1, 0, 1

Newton@Newton-PC-2 ~/C_Program_Design
$

```

上机运行的结果与你分析的结果是否一致？不一致的原因何在？

不一致。b 的第二次输出我觉得是 3，但是动态调试的结果却是在经过 `x=a||b--` 之后，b 的数值不改变

请在程序的最后一个花括号前加上语句：`getchar()`；后重新运行程序，注意事项加上这一语句后，程序的运行进程有何不同？

`getchar()`是一个函数调用，其作用是等待接收你从键盘输入的一个字符，在你未按键之前一直处于等待状态。在这里可以起到暂停的作用。当你看清结果并按任意键后，立即退出程序并切换回 TC 主屏幕。用这一方法可以减少屏幕切换操作。

1.9

下列 C 程序的功能是：计算并输出分段函数值。

$$f(x) = \begin{cases} \frac{1}{x+2}, & -5 \leq x < 0 \text{ and } x \neq -2 \\ \frac{1}{x+5}, & 0 \leq x < 5 \\ \frac{1}{x+12}, & 5 \leq x < 10 \\ 0, & \text{esle} \end{cases}$$

根据程序写出分段函数。其中 x 由键盘输入。请通过调试修改程序中的错误(包括语法错误和逻辑错误)。具体要求如下：

- ①不允许改变计算的精度。
- ②不允许改变原来程序的结构，只能在语句或表达式内部进行修改。
- ③调试正确后，用 $x = -7.0, -2.0, -1.0, 0.0, 2.0, 5.0, 8.0, 10.0, 11.0$ 运行这个程序。
- ④画出与调试正确后的程序对应的流程图。

1.9.1 程序代码

```

1  /*
2  * filename: 3.12 sectioned function.c
3  * property: example
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      float x, y;
10     printf("input x = ");
11     scanf("%f", &x);
12     if((-5.0 <= x) && (x < 0.0) && (x != -2.0)) {
13         y = 1.0 / (x + 2.0);
14     }
15     else {
16         if((x < 5.0) && (x >= 0)) {
17             y = 1.0 / (x + 5.0);
18         }
19         else {
20             if(x < 10.0 && (x >= 5.0) ) {
21                 y = 1.0 / (x + 12.0);
22             }
23             else {
24                 y = 0.0;

```

```

25     }
26 }
27 }
28 printf("x = %e\n y = %e\n", x, y);
29 return 0;
30 }

```

1.9.2 运行结果。

```

~/_C_Program_Design
Newton@Newton-PC-2 ~/_C_Program_Design
$ gcc 3.12\ sectioned\ function.c -o 3.12.exe
Newton@Newton-PC-2 ~/_C_Program_Design
$ ./3.12.exe
input x = 369
x = 3.690000e+02
y = 0.000000e+00
Newton@Newton-PC-2 ~/_C_Program_Design
$

```

1.10 成绩等级划分

用 `scanf` 函数输入一个百分制成绩(整型量), 要求输出成绩等级 A, B, C, D, E。其中 90~100 分为 A, 80~89 分为 B, 70~79 分为 C, 60~69 分为 D, 60 分以下为 E。

具体要求如下:

- (1) 用 `if` 语句实现分支。
- (2) 在输入百分制成绩前要有提示。
- (3) 在输入百分制成绩后, 要判断该成绩的合理性, 对于不合理的成绩应输出出错信息。
- (4) 在输出结果中应包括百分制成绩与成绩等级, 并要有文字说明。
- (5) 分别输入百分制成绩: -90, 100, 90, 85, 70, 60, 45, 101, 运行该程序。

1.10.1 程序代码

```

1  /*
2  * filename: 3.13 grade class.c
3  * property: homework
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      int score;
10     char grade;
11
12     printf("Plesase input the grade (0~100): ");
13     scanf("%d", &score);
14     if(score >= 90 && score <= 100)
15         grade = 'A';
16     else
17         if(score >= 80 && score <= 89)
18             grade = 'B';

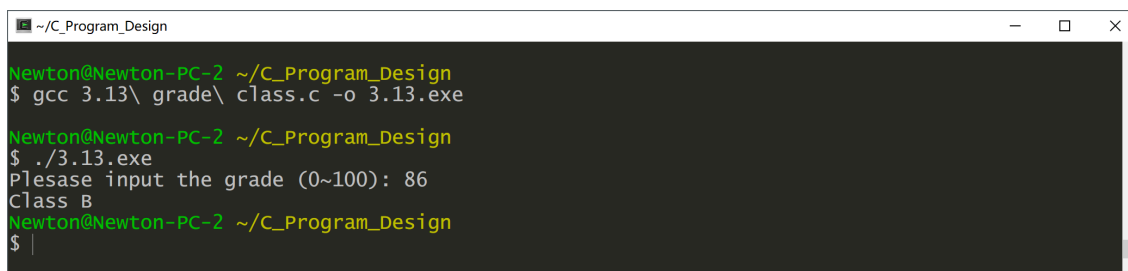
```

```

19     else
20         if(score >= 70 && score <= 79)
21             grade = 'C';
22         else
23             if(score >= 60 && score <= 69)
24                 grade = 'D';
25             else
26                 printf("Invalid input!");
27     printf("Class %c", grade);
28     return 0;
29 }

```

1.10.2 运行结果



```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.13\ grade\ class.c -o 3.13.exe
Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.13.exe
Plesase input the grade (0~100): 86
Class B
Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.11 寻找序列最值的最值

编程找出 5 个整数中的最大数和最小数，并输出找到的最大数和最小数。

1.11.1 程序代码

```

1  /*
2  * filename: 3.14 find max and min.cpp
3  * property: exercise
4  */
5
6  #include<iostream>
7  #include<algorithm>
8
9  using namespace std;
10
11 int main() {
12     int a[5] = {9, 6, 3, 8, 5};
13     for(int i = 0; i < 5; i++)
14         cout << a[i] << " ";
15
16     cout << endl;
17
18     sort(a, a + 5);
19     for(int i = 0; i < 5; i++)
20         cout << a[i] << " ";
21     cout << endl << "max = " << a[4];
22     cout << endl << "min = " << a[0];

```

```

23     return 0;
24 }

```

1.11.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ g++ 3.14\ find\ max\ and\ min.c -o 3.14.exe
Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.14.exe
9 6 3 8 5
3 5 6 8 9
max = 9
min = 3
Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.12 习题

1. 以下 if 语句的形式哪些是错误的？

- (1) `if(x != y)`
- (2) `if(x == y)`
- (3) `if(x > y)`
 then `z = x;`
 else `z = y;`
- (4) `if(x > y)`
 `if(x > z) if(x > m) max = x;`
- (5) `if(a=b)`
 `printf("Yes");`
 else
 `printf("No");`
- (6) `if(5)`
 `x=5;`
 else
 `y=5;`
- (7) `if(x-y)`
 `z = 0;`
 else
 `z=1;`
- (8) `if(x > 0)`
 `y = 0;`
 else `y = 1;`
 else `y = -1;`

2 下面是计算函数

$$f(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ 0, & x < 0 \end{cases}$$

的几个程序段，其中是否存在错误？若有，如何纠正？

(1)

```

1  y=0;
2  if(x <= 0)
3  {
4      if(x < 0) y = -1;

```

```

5     else y = 1;
6 }

```

(2)

```

1 if(x <= 0)
2 if(x < 0) y = -1;
3 else y = 0;
4 else y = 1;

```

(3)

```

1 y = 1;
2 if(x<=0)
3     if(x=0)
4         y=0;
5 else
6     y=-1;

```

(4)

```

1 y = -1;
2 if(x >= 0)
3     if(x > 0)
4         y = 1;
5 else
6     y = 0;

```

1.13 冒泡排序

从键盘输入任意 20 个整数，按小到大的顺序输出。

1.13.1 程序代码

```

1  /*
2   * filename: 3.15 bubble sort.c
3   * property: exercise
4   */
5
6  #include <stdio.h>
7
8  int main() {
9      int a[20];
10     int i, j, t;
11     printf("Please input 20 integers: \n");
12     for(i = 0; i < 20; i++)
13         scanf("%d", &a[i]);
14     printf("\n");
15     for(j = 0; j < 20; j++)
16         for(i = 0; i < 20 - j; i++)
17             if(a[i] > a[i+1]) {

```



```

18         t = a[i];
19         a[i] = a[i+1];
20         a[i+1] = t;
21     }
22     printf("sorted array: \n");
23     for(i=0; i < 20; i++)
24         printf("%d ", a[i]);
25     printf("\n");
26     return 0;
27 }

```

1.13.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.15\ bubble\ sort.c -o 3.15.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.15.exe
Please input 20 integers:
1 25 32 14 5
9 22 12 45 6
0 36 9 25 3
8 24 25 13 3

sorted array:
0 0 1 3 3 5 6 8 9 9 12 13 14 22 24 25 25 32 36

Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.14 大小写转换

输入一个字符，如果是大写字母改变为小写字母；如果是小写字母，则把它变为大写字母；若是其它字符则不变。

1.14.1 程序代码

```

1  /*
2  * filename: 3.16 alphabet.c
3  * property: exercise
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      char ch;
10     scanf("%c", &ch);
11     if(('A' <= ch && ch <= 'Z') || ('a' <= ch && ch <= 'z')) {
12         if('A' <= ch && ch <= 'Z')
13             ch = ch + 32;
14         else
15             ch = ch - 32;
16     }
17     else
18         printf("Invalid input\n");
19     printf("%c\n",ch);

```

```

20     return 0;
21 }

```

1.14.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.16\ alphabet.c -o 3.16.exe
Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.16.exe
K
k
Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.16.exe
a
A
Newton@Newton-PC-2 ~/C_Program_Design
$

```

1.15 指定运算符运算

输入两个数 x 和 y ，以及一个符号 c ，若为‘+’，‘-’，‘*’，‘/’，则输出 $x + y$ ， $x - y$ ， $x \times y$ ， $x \div y$ ，若 c 是其它符号，则输出错误信息。

1.15.1 程序代码

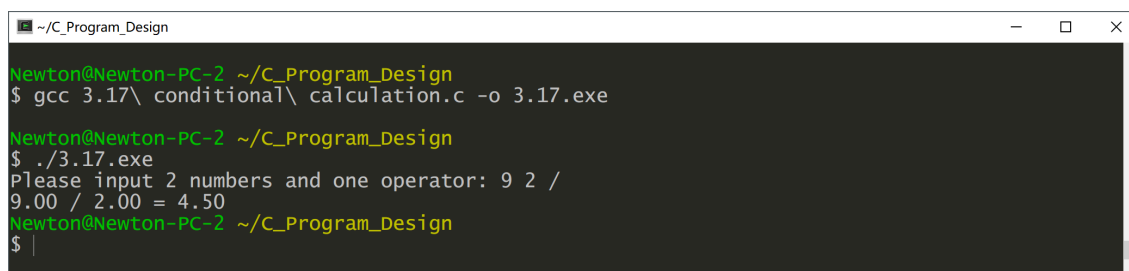
```

1  /*
2  * filename: 3.17 conditional calculation.c
3  * property:
4  */
5
6  #include <stdio.h>
7
8  int main() {
9      float x, y;
10     printf("Please input 2 numbers and one operator: ");
11     char ch;
12     scanf("%f %f %c", &x, &y, &ch);
13
14     if(y == 0 && ch == '/') {
15         printf("Invalid input.\n");
16         return -1;
17     }
18
19     float ans;
20     switch(ch) {
21         case '+': ans = x + y; break;
22         case '-': ans = x - y; break;
23         case '*': ans = x * y; break;
24         case '/': ans = x / y; break;
25         default: printf("error inport!");
26     }

```

```
27     printf("%3.2f %c %3.2f = %3.2f", x, ch, y, ans);
28
29     return 0;
30 }
```

1.15.2 运行结果



```
~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 3.17\ conditional\ calculation.c -o 3.17.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./3.17.exe
Please input 2 numbers and one operator: 9 2 /
9.00 / 2.00 = 4.50
Newton@Newton-PC-2 ~/C_Program_Design
$
```

四、 实验总结

这里最主要的问题就是对于算法的忽视，没能做到先设计算法，再画框图，最后写代码。没能仔细看书，所以因为缺乏一些知识导致了一些问题。代码不严格，刚写完不检查就编译，导致报错太多。要养成良好的程序设计习惯，从一定顺序做起，不能忽视细节[1]。

五、 参考文献

1. 林锐, *高质量 C++/C 编程指南*. 1.0 ed. 2001.

六、 教师评语