程序设计基础及语言

实验指导手册

东南大学
计算机科学与工程学院
软件学院

2011 年 9 月
## 目 录

第一部分 实验环境的使用方法

1.1 目的 ......................................................................................................................... 1

1.2 Visual C++2005 Express Edition Manual ................................................................. 1

1.2.1 First Run ............................................................................................................... 1

1.2.2 New project ......................................................................................................... 3

1.2.3 Add new source file ........................................................................................... 6

1.2.4 Run Program in Dos Prompt ............................................................................... 10

1.2.5 Run program in IDE .......................................................................................... 12

1.2.6 用 Ctrl+F5 运行 ............................................................................................... 16

1.2.7 First Debug ....................................................................................................... 17

第二部分 实验作业及课后习题

Lab1 Introduction to C++ Programming .................................................................... 24

Lab2 Introduction to Classes and Objects .................................................................. 28

Lab3 Control Statements: Part I .................................................................................. 30

Lab4 Control Statements II ......................................................................................... 31

Lab5 Function and Recursion ...................................................................................... 36

Lab6 Array I ................................................................................................................ 44

Lab7 Array II ............................................................................................................... 51

Lab8 Pointers and Pointer-based Strings .................................................................... 54
第一部分 实验环境的使用方法

Visual C++ 2005（简称 VC++）是 Microsoft 公司开发的基于 C/C++的集成开发工具，它是 Visual Studio 中功能强大、代码效率最高的开发工具。

利用 Visual C++可以采用两种方式编写 Win32 应用程序：
第一种：基于 Windows API 的编程方式（又称为控制台编程方式）。
第二种：基于 MFC 的 C++编程方式。

本节介绍 VC++语言的最基本的编程环境和控制台编程方式，通过简单实例介绍 VC++的基本使用方法，使学生尽快学会和掌握如何在 VC++环境下，采用控制台编程方式进行课程实验。

1.1 目的

1. 了解 Visual C++ 2005 的特点。
2. 熟悉 Visual C++ 2005 的控制台开发环境。
3. 学习和掌握 Visual C++ 2005 的编写控制台程序的方法


Created with a trial version of ScreenSteps

1.2.1 First Run

点击 windows 左下角，开始 - 程序，找到新安装的 Microsoft Visual Studio 2005 Express。
点击第二个图标选项，就可以启动 Visual C++ 2005Express。

顺利的话，会看到这样的界面，如果没有看见，一定是安装过程出问题了。
点击右上角小叉，将小窗口关闭。

1.2.2 New project

点击左上角，New Project，新建项目。
选择 emptyproj。

鼠标右击 Source File->Add->Class
1.2.3 Add new source file

或者 Source File->Add->New Item 这两种方法都可以用来添加新项目。这里选择新建项。
弹出此窗口后，选择 C++File，新建一个 cpp 文件，名为 testA，最后选择 Add。

在这个界面里，就可以编辑 testA.cpp 这个文件了。
填写：

```cpp
#include<iostream>

void main() {
    std::cout<<"Hello, My first CPP program.";
}
```
生成解决方案，Build Solution F7。

正常就可以看到，编译成功的消息。
如果说你的源程序有明显的语法错误，按 F7 编译，则会提示编译失败。

编译出来的结果放在这里了。这个路径信息很重要。
1.2.4 Run Program in Dos Prompt

点击开始->运行->输入 cmd,按回车键。

输入以上命令，cd 是更改路径的命令。当然，可以用复制+粘贴命令，把刚才的路径信息复制下来，如果不会就认真输入。
争取变更路径后，用 dir 命令，可以显示当前目录中的文件。确认刚才编译的结果存在。如果不存在，可能是没有编译成功，或者路径错误。

输入 testA.exe，回车键执行该程序。可以看到 Hello 信息，表面程序可以正常的编译执行。
如果想偷懒，在 DOS 窗口下，直接复制路径和执行程序名，会出现错误。

1.2.5 Run program in IDE

当然，大家也可以用 F5，利用调试方式去运行一个程序。
程序不能正常运行。
右击 `testA` 选择 `properties`
Linker->SubSystem  选择下拉框里的 Console
1.2.6 用 Ctrl+F5 运行

接下来，用 Ctrl + F5，来执行程序，就可以看到希望的黑色窗口了。注意 如果使用 F5 来执行还是看不见运行结果的。
两种执行方式的区别在于：
F5 run with debug
Ctrl + F5 run without debug.

如果你要自己看运行结果就用 Ctrl+F5
如果想 Debug 你的 code，看错误出在哪就用 F5。
你可以再 code 最左边你想要开始的 debug 的地方设一个 breakpoint，当你按 F5，你的程序会跳到那里，你可以用 F10 或者 F11 来查看，每行代码的运行结果
1.2.7 First Debug

学习了解 Debug 功能。修改源代码为:

```cpp
#include<iostream>

void main() {
    int i = 0;
    for (i = 0; i<10; i++) {
        std::cout<<"Hello, time(s):"<<i<< std::endl;
    }
}
```

选择 `std::cout<<"Hello, time(s):"<<i<< std::endl;` 这句话。
在菜单中选择 Debug->Toggle Breakpoint，当然，直接按 F9 也可以。

则在这句话前面出现了一个 红色 的标记，称为 断点。
调试之前需要更改一些属性。
Linker->Debugger->Generate Debug Info 改为 Yes。

C/C++->General->Debug Information Format->ZI
C/C++->Optimization->Optimization->Disabled

在菜单中选择 调试 -- 启动调试，当然，直接按 F5 也可以。
程序执行到断点所在位置并停顿下来，下面的自动窗口，给出了代码中变量的值，可以看见变量 i = 0.
可以按绿色按钮（F5）继续直到再次执行到断点，也可以按 F10，按照过程跟踪。这里选择 F5，继续执行，直到下个断点。当程序再次停顿下来时，变量 i=1，已经完成了一次循环。程序中可以设置多个断点，关于 F5，F10，F11 的区别，请进一步学习。
Lab 1 Introduction to C++ Programming

Objectivities:
1. To write simple computer programs in C++.
2. To write simple input and output statements.
3. To use fundamental types.
4. Basic computer memory concepts.
5. To use arithmetic operators.
6. The precedence of arithmetic operators.
7. To write simple decision-making statements

Experiment

- **Ex1:**
  输入书上 p38 例 fig02_03.cpp，熟悉编程环境。

- **Ex2:**
  Description of the Problem
  Write a program that inputs three integers from the keyboard, and prints the sum, average, product, smallest and largest of these numbers. The screen dialogue should appear as follows: [Note: 13, 27 and 14 are input by the user.]

  Sample Output
  
  Input three different integers: **13 27 14**
  Sum is **54**
  Average is **18**
  Product is **4914**
  Smallest is **13**
  Largest is **27**

  Template
  
  ```cpp
  //EX2: numbercompare.cpp
  2 #include <iostream> // allows program to perform input and output
  3
  4 using std::cout; // program uses cout
  5 using std::endl; // program uses endl
  ```
```cpp
using std::cin; // program uses cin

int main()
{
    int number1; // first integer read from user
    int number2; // second integer read from user
    int number3; // third integer read from user
    int smallest; // smallest integer read from user
    int largest; // largest integer read from user

    cout << "Input three different integers: "; // prompt

    /* Write a statement to read in values for number1, number2 and number3 using a single cin statement */

    largest = number1; // assume first integer is largest

    /* Write a statement to determine if number2 is greater than largest. If so assign number2 to largest */

    /* Write a statement to determine if number3 is greater than largest. If so assign number3 to largest */

    smallest = number1; // assume first integer is smallest

    /* Write a statement to determine if number2 is less than smallest. If so assign number2 to smallest */

    /* Write a statement to determine if number3 is less than smallest. If so assign number3 to smallest */

    /* Write an output statement that prints the sum, average, product, largest and smallest */

    return 0; // indicate successful termination
}
```

**Problem-Solving Tips**

1. Prompt the user to input three integer values. You will use a single cin statement to read all three values.
2. Sometimes it is useful to make an assumption to help solve or simplify a problem. For example, we assume number1 is the largest of the three values and assign it to largest. You will use if statements to determine whether number2 or number3 are
larger.
3. Using an if statement, compare largest to number2. If the content of number2 is larger, then store the variable’s value in largest.
4. Using an if statement, compare largest to number3. If the content of number3 is larger, then store the variable’s value in largest. At this point you are guaranteed to have the largest value stored in largest.
5. Perform similar steps to those in Steps 2–4 to determine the smallest value.
6. Write a cout statement that outputs the sum, average, product (i.e., multiplication), largest and smallest values.
7. Be sure to follow the spacing and indentation conventions mentioned in the text.
8. If you have any questions as you proceed, ask your lab instructor for assistance.

Follow-Up Questions and Activities
1. Modify your solution to use three separate cin statements rather than one. Write a separate prompt for each cin.

2. Does it matter whether < or <= is used when making comparisons to determine the smallest integer? Which did you use and why?

● Ex3:

Description of the Problem
Write a program that reads in two integers and determines and prints whether the first is a multiple of the second.[Hint: Use the modulus operator.]

Sample Output
Enter two integers: 22 8
22 is not a multiple of 8

Problem-Solving Tips
1. The input data consists of two integers, so you will need two int variables to store the input values.
2. Use cin to read the user input into the int variables.
3. Use an if statement to determine whether the first number input is a multiple of the second number input. Use the modulus operator, %. If one number divides into another evenly, the modulus operation results in 0. If the result is 0, display a message indicating that the first number is a multiple of the second number.
4. Use an if statement to determine whether the first number input is not a multiple of the second number input. If one number does not divide into another evenly, the modulus operation results in a non-zero value. If non-zero, display a message indicating that the first number is not a multiple of the second.
5. Be sure to follow the spacing and indentation conventions mentioned in the text.
6. If you have any questions as you proceed, ask your lab instructor for assistance.

● Ex4:
**Problem Description**
Write a program that inputs a five-digit number, separates the number into its individual digits and prints the digits separated from one another by three spaces each.

[Hint: Use integer division and the modulus operator.] For example, if the user inputs 42339, the program should print what is shown in the sample output.

**Sample Output**

```
4 2 3 3 9
```

**Template**

```cpp
// ex4
#include <iostream> // allows program to perform input and output

using std::cout; // program uses cout
using std::endl; // program uses endl
using std::cin; // program uses cin

int main()
{
    int number; // integer read from user

    cout << "Enter a five-digit integer: "; // prompt
    cin >> number; // read integer from user

    /* Write a statement to print the left-most digit of the 5-digit number */
    /* Write a statement that changes number from 5-digits to 4-digits */
    /* Write a statement to print the left-most digit of the 4-digit number */
    /* Write a statement that changes number from 4-digits to 3-digits */
    /* Write a statement to print the left-most digit of the 3-digit number */
    /* Write a statement that changes number from 3-digits to 2-digits */
    /* Write a statement to print the left-most digit of the 2-digit number */
    /* Write a statement that changes number from 2-digits to 1-digit */
    cout << number << endl;

    return 0; // indicate successful termination
```
Problem-Solving Tips

1. The input data consists of one integer, so you will use an int variable (number) to represent it. Note that the description indicates that one five-digit number is to be input—not five separate digits.

2. You will use a series of statements to “break down” the number into its individual digits using modulus (%) and division (/) calculations.

3. After the number has been input using cin, divide the number by 10000 to get the leftmost digit. Why does this work? In C++, dividing an integer by an integer results in an integer. For example, 42339 /10000 evaluates to 4 because 10000 divides evenly into 42339 four times. The remainder 2339 is truncated.

4. Change the number to a 4-digit number using the modulus operator. The number modulus 10000 evaluates to the integer remainder—in this case, the right-most four digits. For example, 42339 % 10000 results in 2339. Assign the result of this modulus operation to the variable that stores the five-digit number input.

5. Repeat this pattern of division and modulus reducing the divisor by a factor of 10 each time (i.e., 1000, 100, 10). After the number is changed to a four-digit number, divide/modulus by 100. After the number is changed to a three-digit number, divide/modulus by 100. And so on.

6. Be sure to follow the spacing and indentation conventions mentioned in the text.

7. If you have any questions as you proceed, ask your lab instructor for assistance.

Follow-Up Questions and Activities

1. What happens when the user inputs a number which has fewer than five digits? Why? What is the output when 1763 is entered?

2. The program you completed in this lab exercise inputs a number with multiple digits and separates the digits. Write the inverse program, a program which asks the user for three one-digit numbers and combines them into a single three-digit number. [Hint: Use multiplication and addition to form the three-digit number.]

Homework

2.10
2.24
2.27

Lab2 Introduction to Classes and Objects

Objectivities

1. How to define a class and use it to create an object

2. How to define member functions in a class to implement the class’ s behaviors.

3. How to declare data members in a class to implement the class’ s attributes.
4. How to call a member function of an object to make that member function perform its task.
5. The differences between data members of a class and local variables of a function.
6. How to use a constructor to ensure that an object’s data is initialized when the object is created.
7. How to engineer a class to separate its interface from its implementation and encourage reuse.

Experiments

- **Ex1(p.99--3.11)**

Description of the Problem
Modify class GradeBook (Figs. 3.11 and Figs. 3.12). Include a second string data member that represents the name of the course’s instructor. Provide a `set` function to change the instructor’s name and a `get` function to retrieve it. Modify the constructor to specify two parameters—one for the course name and one for the instructor’s name. Modify member function `displayMessage` such that it first outputs the welcome message and course name, then outputs "This course is presented by: " followed by the instructor’s name. Modify the test application to demonstrate the class’s new capabilities.

Sample Output

Welcome to the grade book for
CS101 Introduction to C++ Programming!
This course is presented by: Sam Smith
Changing instructor name to Judy Jones
Welcome to the grade book for
CS101 Introduction to C++ Programming!
This course is presented by: Judy Jones

Problem-Solving Tips
1. In class GradeBook, declare a string data member to represent the instructor’s name.
2. Declare a public `set` function for the instructor’s name that does not return a value and takes a string as a parameter. In the body of the `set` function, assign the parameter’s value to the data member that represents the instructor’s name.
3. Declare a public `get` function that returns a string and takes no parameters. This member function should return the instructor’s name.
4. Modify the constructor to take two string parameters. Assign the parameter that represents the instructor’s name to the appropriate data member.
5. Add a `cout` statement to member function `displayMessage` to output the value of the data member you declared earlier.
6. Be sure to follow the spacing and indentation conventions mentioned in the text.
7. If you have any questions as you proceed, ask your lab instructor for help.

- **Ex2(p.100--3.14)**
Sample Output

<table>
<thead>
<tr>
<th>Employee 1: Bob Jones; Yearly Salary: 34500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee 2: Susan Baker; Yearly Salary: 37800</td>
</tr>
<tr>
<td>Increasing employee salaries by 10%</td>
</tr>
<tr>
<td>Employee 1: Bob Jones; Yearly Salary: 37944</td>
</tr>
<tr>
<td>Employee 2: Susan Baker; Yearly Salary: 41580</td>
</tr>
</tbody>
</table>

Problem-Solving Tips
1. Class Employee should declare three data members.
2. The constructor must declare three parameters, one for each data member. The value for the salary should be validated to ensure it is not negative.
3. Declare a public set and get functions for each data member. The set functions should not return values and should each specify a parameter of a type that matches the corresponding data member (string for first name and last name, int for the salary). The get functions should receive no parameters and should specify a return type that matches the corresponding data member.
4. When you call the constructor from the main function, you must pass it three arguments that match the parameters declared by the constructor.
5. Giving each employee a raise will require a call to the get function for the salary to obtain the current salary and a call to the set function for the salary to specify the new salary.
6. Be sure to follow the spacing and indentation conventions mentioned in the text.
7. If you have any questions as you proceed, ask your lab instructor for help.

Homework
3.5
3.13
3.15

Lab3 Control Statements: Part I

Objectivities:
1. Basic problem-solving techniques.
2. To develop algorithm through the process of top-down, stepwise refinement.
3. Counter-controlled repetition and sentinel-controlled repetition.
4. To use the if and if...else selection statements to choose among alternative actions.
5. To use the while repetition statement to execute statements in a program repeatedly.

Experiment
- Ex1: Problem
(P142) 4.17 The process of finding the largest number (i.e., the maximum of a group of numbers) is used frequently in computer applications. For example, a program that determines the winner of a sales contest inputs the number of units sold by each salesperson. The salesperson who sells the most units wins the contest.
Your program should use three variables, as follows:

- **counter**: A counter to count to 10 (i.e., to keep track of how many numbers have been input and to determine when all 10 numbers have been processed).
- **number**: The current number input to the program.
- **largest**: The largest number found so far.

✔ **Content and Requirement:**

1. Write a pseudocode program.
2. Write a C++ program
   a) To input 10 numbers from keyboard by the user.
   b) To use a `while` and `if` statement to determine the largest number.
   c) To print the largest number.
3. Execute the program.

✔ **Ex2. Problem**

(P145) 4.26 A Palindrome is a number or text phrase that reads the same backward as forward. For example, each of the following five-digit integers is a palindrome: 12321, 55555, 45554 and 11611.

✔ **Content and Requirement:**

1. Write a C++ program
   a) To reads in a five-digit integer.
   b) To use a `while` and `if…else` statement to determine whether it is a palindrome.
   c) To Output “the five-digit integers is a palindrome!” or “It is not a palindrome!”.
2. Execute the program.

✔ **Hint:**

*Use the division and modulus operators to separate the number into its individual digits.*

**Homework**

4.27, 4.34, 4.35(b)

**Lab4 Control Statements II**

**Objective**

- The essentials of counter-controlled repetition.
- To use the `for` and `do…while` repetition statements to execute statements in a program repeatedly.
- To understand multiple selection using the `switch` selection statement.
- To use the `break` and `continue` program control statements to alter the flow of control.
To use the logical operators to form complex conditional expressions in control statements.
To avoid the consequences of confusing the equality and assignment operators.

Experiments

- Ex 1: Integer Average (课后习题5.6)

Lab Objectives
- Using sentinel-controlled repetition with a for loop.
The follow-up question and activity also will give you practice:
- Using counter-controlled repetition with a for loop.

Description of the Problem
Write a program that uses a for statement to calculate and print the average of several integers. Assume the last value read is the sentinel 9999. A typical input sequence might be 10 8 11 7 9 9999 indicating that the program should calculate the average of all the values preceding 9999.

Sample Output

Enter integers (9999 to end):
10 8 11 7 9 9999
The average is: 9

Template

```cpp
1 // Lab 1: IntegerAverage.cpp
2 // Calculate the average of several integers.
3
4 #include <iostream>
5 using std::cin;
6 using std::cout;
7 using std::endl;
8
9 int main()
10 {
11 int value; // current value
12 int count = 0; // number of inputs
13 int total; // sum of inputs
14
15 // prompt for input
16 cout << "Enter integers (9999 to end):" << endl;
17 cin >> value;
18```

32
Problem-Solving Tips
1. When used for sentinel-controlled repetition, a for loop can be written much like a while loop, using the same loop-continuation condition as a while loop.
2. When performing sentinel-controlled repetition, a for loop does not need to increment any counter variable. But it can still initialize a variable if so desired.

Follow-up Question
1. Modify the program to perform counter-controlled repetition with a for loop. Assume that the first integer entered by the user represents the number of subsequent integers that the user will input to be averaged.

   Sample output:

<table>
<thead>
<tr>
<th>Enter integers (first integer should be the number of subsequent integers):</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 12 7 4 19 6</td>
</tr>
</tbody>
</table>

• Ex 2: Pythagorean Triples （课后习题5.20）

Lab Objectives
• Using counter-controlled repetition.
• Using “brute force” to solve a problem.
• Nesting for loops.
The follow-up questions and activities will also give you practice:
• Using break statements.
• Using continue statements.
• Using long integers.

**Description of the Problem**
A right triangle can have sides that are all integers. A set of three integer values for the sides of a right triangle is called a Pythagorean triple. These three sides must satisfy the relationship that the sum of the squares of two of the sides is equal to the square of the hypotenuse. Find all Pythagorean triples for side1, side2 and hypotenuse all no larger than 500. Use a triple-nested for loop that tries all possibilities. This is an example of “brute force computing.” You will learn in more advanced computer-science courses that there are many interesting problems for which there is no known algorithmic approach other than using sheer brute force.

**Sample Output**

<table>
<thead>
<tr>
<th>Side 1</th>
<th>Side 2</th>
<th>Side 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>24</td>
<td>25</td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
</tr>
<tr>
<td>319</td>
<td>360</td>
<td>481</td>
</tr>
<tr>
<td>320</td>
<td>336</td>
<td>464</td>
</tr>
<tr>
<td>325</td>
<td>360</td>
<td>485</td>
</tr>
<tr>
<td>340</td>
<td>357</td>
<td>493</td>
</tr>
</tbody>
</table>

A total of 386 triples were found.

**Template**

```cpp
#include <iostream>

using std::cout;
using std::endl;

int main()
{
    int count = 0; // number of triples found
    long int hypotenuseSquared; // hypotenuse squared
    long int sidesSquared; // sum of squares of sides

    cout << "Side 1\tSide 2\tSide3" << endl;
    // side1 values range from 1 to 500
    /* Write a for header for side1 */
    {
        // side2 values range from current side1 to 500
    }
    // side2 values range from current side1 to 500
    ```
19 /* Write a for header for side2 */
20 {
21     // hypotenuse values range from current side2 to 500
22     /* Write a for header for hypotenuse */
23     {
24         // calculate square of hypotenuse value
25         /* Write a statement to calculate hypotenuseSquared */
26         // calculate sum of squares of sides
27         /* Write a statement to calculate the sum of the sides Squared */
28         // if (hypotenuse)^2 = (side1)^2 + (side2)^2,
29         // Pythagorean triple
30         if ( hypotenuseSquared == sidesSquared )
31             {
32                 // display triple
33                 cout << side1 << " \t" << side2 << " \t"
34                     << hypotenuse << " \n";
35                 count++; // update count
36             } // end if
37         } // end for
38     } // end for
39 } // end for
40
41 // display total number of triples found
42 cout << "A total of " << count << " triples were found."
43 return 0; // indicate successful termination
44 } // end main

Problem-Solving Tips
1. This program does not require any input from the user.
2. This program can take a significant amount of time to run, depending on your computer’s processor speed. If you have a CPU monitor available on your system, it is worth taking a look at it when this program executes.
3. Do not be concerned that you are trying values that do not seem to make sense, such as a 1–1–500 triangle.
4. Remember that brute-force techniques try all possible values.
5. The formula for the Pythagorean Theorem is $\text{hypotenuse}^2 = (\text{side 1})^2 + (\text{side 2})^2$.
6. To avoid producing duplicate Pythagorean triples, start the second for loop at $\text{side2} = \text{side1}$ and the third for loop at $\text{hypotenuse} = \text{side2}$. This way, when a Pythagorean triple is found, side1 will be the shortest side of the triangle and hypotenuse will be the longest side.

Follow-up Question
1. How many times did this program execute the innermost for loop? Add another counter to the program that counts the number of times this loop iterates. Declare a new variable of type **long**, named `loopCounter` and initialize it to 0. Then add a statement in the innermost for statement that increments `loopCounter` by 1. Before exiting the program, print the value of `loopCounter`. Do the numbers match?

2. Add a `break` statement to the program inside the innermost for loop. This `break` statement should be called after the 20th Pythagorean triple is found. Explain what happens to the program after the 20th triple is found. Are all three for loops exited, or just the innermost one? What happens when the `break` statement is placed inside the middle loop? The outermost loop?

3. Add a `continue` statement to the program that prevents a Pythagorean triple from being found when `side1` is equal to 8. Using your solution to **Follow-Up Question 1**, calculate how many times this new program executes the innermost for loop. Explain why the `continue` statement affected the output.

4. Explain why a `long` variable is used for `hypotenuseSquared` and `sideSquared`. Modify the program so that they are both of type `short` instead of type `long`. Rerun the program. What happens?

**Homework**

Ex 5.8, Ex5.9 (for loop),
Ex 5.14 (for / while and switch statements),
Ex 5.17, Ex 5.22 (logical operators),
Ex 5.7, Ex 5.26, Ex 5.12 (nested for loops)
Ex5.23

**Lab5 Function and Recursion**

**Objective**

1. 掌握模块化构建程序的方法
2. 掌握函数定义、函数声明、参数传递的方法
3. 掌握利用随机数生成机制实现模拟技术
4. 编写 C++程序：函数调用和递归调用
5. 理解作用域的概念，以及标识符是如何被限制在程序的特定区域中
6. 理解参数传递中传值调用与传引用调用之间的区别，何时应用传值调用，何时应用传引用调用

**Experiments**

- 1. 习题 6.24
  问题描述
Use techniques similar to those developed in 6.22 and 6.23 to produce a program that graphs a wide range of shapes.

分别用三个函数实现绘制正方形(square)、菱形(diamond)、三角形(triangle)，在 main 函数中用循环结构嵌套 switch 结构实现对各种图形函数的调用测试。

实例输出

Choose the shape to graph
1 for square
2 for diamond
3 for triangle.
? 2
Enter a character and size: 5

Do you want to continue (y or n)?: y

Choose the shape to graph
1 for square
2 for diamond
3 for triangle.
? 1
Enter a character and size: 6

Do you want to continue (y or n)?: n

2. (习题 6.29):

问题描述

An integer is said to be a perfect number if the sum of its factors, including 1 (but not the number itself), is equal to the number. For example, 6 is a perfect number, because $6 = 1 + 2 + 3$. Write a function perfect that determines whether parameter number is a perfect number. Use this function in a program that determines and prints all the perfect numbers between 1 and 1000. Print the factors of each perfect number to confirm that the number is indeed perfect. Challenge the power of your computer by testing numbers much larger than 1000.

一个完全数是这样的一个整数，它的所有因子（包含 1 但不包含该数本身）的和恰好等于该数本身。举个例子，6 是一个完全数，因为 $6 = 1 + 2 + 3$。编写一个函数 perfect(bool 类型返回值)，用它来判断该函数的形式参数是否为一个完全数。在程序中调用该函数，找出 1 至 1000 这个范围内所有的完全数并打印出来。并且将这些找出来的完全数的因子打印出来，用来确认该数确实是一个完全数。你可以通过测试数值在 1000 以上的完全数来挑战
你计算机的能力。

实例输出

```
Perfect integers between 1 and 1000:
  6 = 1 + 2 + 3
 28 = 1 + 2 + 4 + 7 + 14
496 = 1 + 2 + 4 + 8 + 16 + 31 + 62 + 124 + 248
Press any key to continue
```

3. (习题 6.30):

问题描述

(英文) (PrimeNumbers) An integer is said to be prime if it is divisible by only 1 and itself. For example, 2, 3, 5 and 7 are prime, but 4, 6, 8 and 9 are not.
1. Write a function that determines whether a number is prime.
2. Use this function in a program that determines and prints all the prime numbers between 2 and 10,000. How many of these numbers do you really have to test before being sure that you have found all the primes?
3. Initially, you might think that n/2 is the upper limit for which you must test to see whether a number is prime, but you need only go as high as the square root of n. Why? Rewrite the program, and run it both ways. Estimate the performance improvement.

(中文) (质数问题)一个质数是这样的一个整数，它只能被 1 和该数本身整除。举例来说，2, 3, 4, 7 是质数，而 4, 6, 8, 9 不是质数。
1. 编写一个函数来判断一个数是否为质数。
2. 在程序中调用这个函数，找出 2 至 10000 这个范围内所有的质数并且打印出来。在你确认找出所有质数之前你测试了多少次呢？
3. 最初，你可能认为判断一个数是否为质数的测试次数上限为 n/2，但实际你需要的测试次数的上限只是 n 的平方根。这是为什么呢？重新编写这个程序，分别在这两种方式下运行，并观察性能的改进。

实例输出

```
The prime numbers from 1 to 10000 are:
  2  3  5  7 11 13 17 19 23 29
 31 37 41 43 47 53 59 61 67 71
 73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173
179 181 191 193 197 199 211 223 227 229
233 239 241 251 257 263 269 271 277 281
283 293 307 311 313 317 331 337 347 349
353 359 367 373 379 383 389 397 401 409
419 421 431 433 439 443 449 457 461 463
479 487 491 499 503 509 521 523 541
9293 9297 9311 9319 9323 9337 9341 9343 9349 9353
9391 9397 9403 9413 9419 9421 9431 9433 9437 9439
9461 9467 9473 9479 9491 9497 9511 9521 9533 9541
9551 9557 9563 9577 9581 9601 9613 9619 9623 9629
9631 9643 9649 9661 9677 9679 9689 9697 9719 9721
9733 9739 9743 9749 9757 9781 9877 9879 9919 9923
9929 9931 9941 9949 9967 9973
Total of 1229 prime numbers between 1 and 10000.
Press any key to continue
```
4. (习题 6.32):

问题描述

(英文) The greatest common divisor (GCD) of two integers is the largest integer that evenly divides each of the numbers. Write a function GCD that returns the greatest common divisor of two integers.

(中文) 两个整型数的最大公约数(The greatest common divisor, GCD)定义为最大的整除这两个整数的数。编写 gcd 函数，返回两个整型数的最大公约数。

实例输出

Enter two integers: 6 8
The greatest common divisor of 6 and 8 is 2

Enter two integers: 789 4
The greatest common divisor of 789 and 4 is 1

Enter two integers: 9999 27
The greatest common divisor of 9999 and 27 is 9

Enter two integers: 73652 8
The greatest common divisor of 73652 and 8 is 4

Enter two integers: 99 11
The greatest common divisor of 99 and 11 is 11

步骤要求:

(1) 程序需要输入两个整数，编写一个循环语句，以便于程序一次执行就允许用户多次输入测试整数对，是一种十分便捷的测试程序。

(2) 程序中包含一个 gcd 函数，该函数完成最大公约数的求解算法。

提示：最大公约数的求解算法是应用模数运算符来计算余数，如果两个数被除后的余数均为 0，那么，除数就是公约数，然后可以从函数中返回最大公约数。

(3) 采用简单尝试的方法实现 gcd 函数，命名为 gcd0 函数。要求通过引用参数返回最大公约数；函数无返回值；三个参数，其中一个为引用参数，返回最大公约数。

(4) 采用迭代法来实现 gcd 函数，命名为 gcd1 函数。

(5) 利用调试工具，监测程序执行过程中变量的改变。

欧几里德算法提示:

首先用小的数对大的数取余；如果余数是零，那么两个数相同，或者这个较小的数就是它们的最大公约数；如果余数不为零，则继续用这次计算得出的余数对两个整数中比较小的数取余。如果余数为零，则这个比较小的数为最大公约数，如果余数不为零，则继续用本次的余数对上次获得的余数取余，如果余数为零，则上次计算得出的余数则为最大公约数，如果不为零则继续递归，直到得出最大公约数。

<table>
<thead>
<tr>
<th>数 1</th>
<th>数 2</th>
<th>余</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>36</td>
<td>8</td>
</tr>
<tr>
<td>36</td>
<td>8</td>
<td>4</td>
</tr>
</tbody>
</table>
8 | 4 | 0

因此結果即是 4。

5、（题 6.40）

问题描述

(英文) Write a recursive function power ( base, exponent ) that, when invoked, returns base
exponent

For example, power ( 3, 4 ) = 3 * 3 * 3 * 3. Assume that exponent is an integer greater
than or equal to 1. Hint: The recursion step would use the relationship

\[ \text{base}^\text{exponent} = \text{base} \cdot \text{base}^{\text{exponent} - 1} \]

and the terminating condition occurs when exponent is equal to 1, because

\[ \text{base}^1 = \text{base} \]

(中文) 编写一个递归函数 power ( base, exponent )，当函数被调用时，返回

base exponent

例如，power ( 3, 4 ) = 3 * 3 * 3 * 3。假设指数 exponent 是一个大于或等于 0 的整
数。提示：递归步骤使用关系式

\[ \text{base}^\text{exponent} = \text{base} \cdot \text{base}^{\text{exponent} - 1} \]

当指数 exponent = 1 时，递归终止条件发生，因为此时 base^1 = base。

实例输出

Enter a base and an exponent: 3 5
3 raised to the 5 is 243

6、（题 6.40）

问题描述

(英文) (Fibonacci Series) The Fibonacci series

0, 1, 1, 2, 3, 5, 8, 13, 21, ...

begins with the terms 0 and 1 and has the property that each succeeding term is the
sum of the two preceding terms. (a) Write a nonrecursive function fibonacci( n ) that
calculates the nth Fibonacci number. (b) Determine the largest int Fibonacci number
that can be printed on your system. Modify the program of part (a) to use double
instead of int to calculate and return Fibonacci numbers, and use this modified
program to repeat part (b).

(中文) 斐波那契数列

0, 1, 1, 2, 3, 5, 8, 13, 21, ...

从数值 0 和 1 开始，具有每一项斐波那契数列的值是前两项斐波那契数列的值之
和的性质。 (a) 编写一个非递归函数 fibonacci( n ) 计算第 n 项斐波那契数列的值。

(b) 确定能打印输出在你系统中的最大的斐波那契数列的整型值。 修改 (a) 部分的程序，
使用 double 类型的值代替 int 型的值来计算和返回斐波那契数列的值，并且使用这修改过的
程序来实现 (b) 部分的功能。

实例输出

整型输出:

fibonacci( 0 ) = 0
fibonacci( 1 ) = 1

实例输出
7、（习题 6.45）:

问题描述

(英文) (Recursive Greatest Common Divisor) The greatest common divisor of integers x and y is the largest integer that evenly divides both x and y. Write a recursive function gcd that returns the greatest common divisor of x and y, defined recursively as follows: If y is equal to 0, then gcd(x, y) is x; otherwise, gcd(x, y) = gcd(y, x % y), where % is the modulus operator. [Note: For this algorithm, x must be larger than y.]

(中文) (递归法求最大公约数) 整数 x 和 y 的最大公约数是能同时被 x 和 y 整除的最大整数。编写一个函数 gcd()，返回 x 和 y 的最大公约数，定义递归如下：假如 y=0，gcd(x, y) = x；否则，gcd(x, y) = gcd(y, x % y)，这里的%是求模运算符。[注：对于此算法，x 必须大于 y。]

实例输出
Enter two integers: 30 10
Greatest common divisor of 30 and 10 is 10

**Homework**

1. (Exercise 6.27)

**Problem Description**

(Celsius and Fahrenheit Temperatures) Implement the following integer functions: Function `celsius` returns the Celsius equivalent of a Fahrenheit temperature. Function `fahrenheit` returns the Fahrenheit equivalent of a Celsius temperature. Use these functions to write a program that prints charts showing the Fahrenheit equivalents of all Celsius temperatures from 0 to 100 degrees, and the Celsius equivalents of all Fahrenheit temperatures from 32 to 212 degrees. Print the outputs in a neat tabular format that minimizes the number of lines of output while remaining readable.

(Chinese)(摄氏和华氏温度) 实现以下功能（整型）：摄氏函数返回与一个华氏温度等值的摄氏度温度，华氏函数返回与一个摄氏温度等值的华氏度温度。使用这些函数编写一个程序打印输出图表，图表显示与所有摄氏0到100度等值的华氏度温度和与所有华氏32到212度等值的摄氏度温度。以一种在保持可读性的情况下尽可能减少输出的行数的形式打印输出一个排列整齐的表格。

**Example Output**

```
<p>| Fahrenheit equivalents of Celsius temperatures: |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th>Celsius</th>
<th>Fahrenheit</th>
<th>Celsius</th>
<th>Fahrenheit</th>
<th>Celsius</th>
<th>Fahrenheit</th>
<th>Celsius</th>
<th>Fahrenheit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>32</td>
<td>25</td>
<td>??</td>
<td>50</td>
<td>122</td>
<td>??</td>
<td>75</td>
</tr>
<tr>
<td>1</td>
<td>33</td>
<td>26</td>
<td>78</td>
<td>51</td>
<td>123</td>
<td>76</td>
<td>168</td>
</tr>
<tr>
<td>2</td>
<td>35</td>
<td>27</td>
<td>80</td>
<td>52</td>
<td>125</td>
<td>77</td>
<td>170</td>
</tr>
<tr>
<td>3</td>
<td>37</td>
<td>28</td>
<td>82</td>
<td>53</td>
<td>127</td>
<td>78</td>
<td>172</td>
</tr>
<tr>
<td>4</td>
<td>39</td>
<td>29</td>
<td>84</td>
<td>54</td>
<td>129</td>
<td>79</td>
<td>174</td>
</tr>
<tr>
<td>5</td>
<td>41</td>
<td>30</td>
<td>86</td>
<td>55</td>
<td>131</td>
<td>80</td>
<td>176</td>
</tr>
<tr>
<td>6</td>
<td>42</td>
<td>31</td>
<td>87</td>
<td>56</td>
<td>132</td>
<td>81</td>
<td>177</td>
</tr>
<tr>
<td>7</td>
<td>44</td>
<td>32</td>
<td>89</td>
<td>57</td>
<td>134</td>
<td>82</td>
<td>179</td>
</tr>
<tr>
<td>8</td>
<td>46</td>
<td>33</td>
<td>91</td>
<td>58</td>
<td>136</td>
<td>83</td>
<td>181</td>
</tr>
<tr>
<td>9</td>
<td>48</td>
<td>34</td>
<td>93</td>
<td>59</td>
<td>138</td>
<td>84</td>
<td>183</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>35</td>
<td>95</td>
<td>60</td>
<td>140</td>
<td>85</td>
<td>185</td>
</tr>
</tbody>
</table>
```

```
<p>| Celsius equivalents of Fahrenheit temperatures: |
|-----------------|-----------------|-----------------|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th>Fahrenheit</th>
<th>Celsius</th>
<th>Fahrenheit</th>
<th>Celsius</th>
<th>Fahrenheit</th>
<th>Celsius</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>0</td>
<td>77</td>
<td>25</td>
<td>122</td>
<td>50</td>
</tr>
<tr>
<td>33</td>
<td>0</td>
<td>78</td>
<td>25</td>
<td>123</td>
<td>50</td>
</tr>
<tr>
<td>34</td>
<td>1</td>
<td>79</td>
<td>26</td>
<td>124</td>
<td>51</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>80</td>
<td>26</td>
<td>125</td>
<td>51</td>
</tr>
<tr>
<td>36</td>
<td>2</td>
<td>81</td>
<td>27</td>
<td>126</td>
<td>52</td>
</tr>
<tr>
<td>37</td>
<td>2</td>
<td>82</td>
<td>27</td>
<td>127</td>
<td>52</td>
</tr>
<tr>
<td>38</td>
<td>3</td>
<td>83</td>
<td>28</td>
<td>128</td>
<td>53</td>
</tr>
<tr>
<td>39</td>
<td>3</td>
<td>84</td>
<td>28</td>
<td>129</td>
<td>53</td>
</tr>
<tr>
<td>40</td>
<td>4</td>
<td>85</td>
<td>29</td>
<td>130</td>
<td>54</td>
</tr>
<tr>
<td>41</td>
<td>5</td>
<td>86</td>
<td>30</td>
<td>131</td>
<td>55</td>
</tr>
<tr>
<td>42</td>
<td>5</td>
<td>87</td>
<td>30</td>
<td>132</td>
<td>55</td>
</tr>
<tr>
<td>43</td>
<td>6</td>
<td>88</td>
<td>31</td>
<td>133</td>
<td>56</td>
</tr>
</tbody>
</table>
```

2、(习题 6.35 - 37):

问题描述
（1）(中文)计算机在教育领域的作用越来越大，编写一个程序，帮助小学生学习乘法。用 `rand` 函数产生两个一位正整数，然后输入下列问题：

```
How much is 6 times 7?
```

然后学生输入答案，程序检查学生的答案。如果正确，则打印“very good”。然后提出另一个乘法问题。如果不正确，则打印“No. Please try again.”，然后让学生重复回答这个问题，直到答对。

（2）在教学中使用计算机称为计算机辅助教学(CAI)。CAI 环境中出现的一个问题是学生容易疲劳。要消除这个问题，可以改变计算机的对话来保持学生的注意力。修改练习(1)的程序，使每次学生答对时和答错时打印不同的评语。

答对时打印：
```
Very good!
Excellent!
Nice work!
Keep up the good work!
```

Responses to an incorrect answer
```
No. Please try again.
Wrong. Try once more.
Don’t give up!
No. Keep trying.
```

用随机数产生器选择 1 到 4 的数，由此选择相应评语。用 `switch` 结构发出响应。

（3）更复杂的计算机辅助教学(CAI)系统需要监视学生在一段时间的成绩。新内容的推出通常是在学生学好旧内容之后进行的。修改(2)的程序，统计学生答对和答错的比例（每题仅给一次回答机会）。学生答完 10 题后，程序计算其答对率。如果比例不到 75%，则程序打印“Please ask your instructor for extra help”，否则输出“Nice work!”，然后终止。

3、（习题 6.38 - 39）:

问题描述
编写一个猜数字游戏的程序，程序随机选择一个 1 到 1000 的数
```
I have a number between 1 and 1000.
Can you guess my number?
Please type your first guess.
```

然后游戏者输入第一个结果。程序响应如下：
1. Excellent! You guessed the number!
   Would you like to play again (y or n)?
2. Too low. Try again.
3. Too high. Try again.

如果游戏考猜错，则程序进行循环，直到猜对。程序通过 Too high 或 To low 消息帮助学生接近正确答案。说明：这个程序中采用的查找技术称为二分查找。计算游戏者已猜过的次数。如果次数值为 10 以下，则打印 “Either you know the secret or you got lucky!”。如果 10 次猜中，则打印 “Ahak! You know the secret”。如果超过 10 次才猜中，则打印 “You should be able to do better!” 如何在最少的次数内猜中呢？为什么 1 到 1000 的数字能在 10 次之内猜中呢？

4. (习题 6.42): 汉诺塔

Lab6 Array I

Objectivities

1. Using rand to generate random numbers and using srand to seed the random-number generator.
2. Declaring, initializing and referencing arrays.
3. The follow-up questions and activities also will give you practice:
4. Remembering that arrays begin with subscript 0 and recognizing off-by-one errors.
5. Preventing array out-of-bounds errors.
6. Using two-dimensional arrays.

Experiment

Description of the Problem

Write a program that simulates the rolling of two dice. The program should call rand to roll the first die, and should call rand again to roll the second die. The sum of the two values should
then be calculated. [Note: Each die has an integer value from 1 to 6, so the sum of the two values will vary from 2 to 12, with 7 being the most frequent sum and 2 and 12 being the least frequent sums.] Figure L 7.1 shows the 36 possible combinations of the two dice. Your program should roll the two dice 36,000 times. Use a one-dimensional array to tally the numbers of times each sum appears. Print the results in a tabular format. Also, determine if the totals are reasonable (i.e., there are six ways to roll a 7, so approximately one sixth of all the rolls should be 7).

<table>
<thead>
<tr>
<th>Sum</th>
<th>Total</th>
<th>Expected</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1000</td>
<td>2.778%</td>
<td>2.778%</td>
</tr>
<tr>
<td>3</td>
<td>1958</td>
<td>5.556%</td>
<td>5.439%</td>
</tr>
<tr>
<td>4</td>
<td>3048</td>
<td>8.333%</td>
<td>8.467%</td>
</tr>
<tr>
<td>5</td>
<td>3979</td>
<td>11.111%</td>
<td>11.053%</td>
</tr>
<tr>
<td>6</td>
<td>5007</td>
<td>13.889%</td>
<td>13.908%</td>
</tr>
<tr>
<td>7</td>
<td>6087</td>
<td>16.667%</td>
<td>16.908%</td>
</tr>
<tr>
<td>8</td>
<td>4996</td>
<td>13.889%</td>
<td>13.878%</td>
</tr>
<tr>
<td>9</td>
<td>3971</td>
<td>11.111%</td>
<td>11.031%</td>
</tr>
<tr>
<td>10</td>
<td>2996</td>
<td>8.333%</td>
<td>8.322%</td>
</tr>
<tr>
<td>11</td>
<td>2008</td>
<td>5.556%</td>
<td>5.578%</td>
</tr>
<tr>
<td>12</td>
<td>950</td>
<td>2.778%</td>
<td>2.639%</td>
</tr>
</tbody>
</table>

**Template**

```cpp
#include <iostream>
#include <iomanip>
#include <cstdlib>
#include <ctime>

using std::cout;
using std::ios;
using std::setw;
using std::setprecision;
using std::fixed;
using std::showpoint;

const long ROLLS = 36000;
const int SIZE = 13;

int main()
{
    // array expected contains counts for the expected
    // number of times each sum occurs in 36 rolls of the dice
    /* Write the declaration of array expected here. Assign an
    initializer list containing the expected values here. Use
```
SIZE for the number of elements */
int x; // first die
int y; // second die
    /* Write declaration for array sum here. Initialize all
elements to zero. Use SIZE for the number of elements */

srand( time( 0 ) );

    // roll dice 36,000 times
    /* Write a for statement that iterates ROLLS times. Randomly
generate values for x (i.e., die1) and y (i.e, die2)
and increment the appropriate counter in array sum that
corresponds to the sum of x and y */

    cout << setw( 10 ) << "Sum" << setw( 10 ) << "Total" << setw( 10 )
        << "Expected" << setw( 10 ) << "Actual\n" << fixed << showpoint;
    // display results of rolling dice
    for ( int j = 2; j < SIZE; j++ )
    
        cout << setw( 10 ) << j << setw( 10 ) << sum[ j ]
        << setprecision( 3 ) << setw( 9 )
        << 100.0 * expected[ j ] / 36 << "%" << setprecision( 3 )
        << setw( 9 ) << 100.0 * sum[ j ] / 36000 << "%";
    return 0; // indicates successful completion
} // end main

Problem-Solving Tips

1. Remember that array subscripts always begin with zero. This is also true for each dimension of
a multiple-subscripted array (which this lab does not use).

2. The actual percentage is the likelihood, based on the results of your program, that a dice roll
produced a certain result. In other words, if you roll the dice 36,000 times the actual percentage
will be the (number of times a result occurred / 36000) * 100.

3. The expected percentage is the statistical probability that a dice roll will produce a certain result.
This can be calculated from the diagram “36 possible outcomes of rolling two dice,” shown in
the problem description. For example, there is only one combination that will produce the sum of
2 and there are 36 total combinations. Therefore, the expected percentage of rolling a 2 is 1/36 or
2.778%.

Follow-Up Questions and Activities
1. Why is the variable SIZE initialized to 13 when there are only 11 possible die-roll outcomes?

2. What happens if the < operator on line 47 of the program template is changed to <=?

3. What happens if the elements of array sum are not initialized to zero? Try running the program without initializing the array. Show your results.

4. Modify the program to use a two-dimensional array similar to the diagram in Figure L 7.1. Now, rather than counting the number of times each sum appears, increment the correct cell in the array. Print this array with the number of times each dice combination occurred. A sample output may look like the following:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1011</td>
<td>971</td>
<td>1027</td>
<td>1025</td>
<td>971</td>
<td>1015</td>
</tr>
<tr>
<td>2</td>
<td>1013</td>
<td>968</td>
<td>990</td>
<td>968</td>
<td>1081</td>
<td>993</td>
</tr>
<tr>
<td>3</td>
<td>993</td>
<td>1014</td>
<td>983</td>
<td>973</td>
<td>1019</td>
<td>977</td>
</tr>
<tr>
<td>4</td>
<td>980</td>
<td>1004</td>
<td>974</td>
<td>1022</td>
<td>946</td>
<td>1046</td>
</tr>
<tr>
<td>5</td>
<td>1003</td>
<td>1021</td>
<td>1019</td>
<td>979</td>
<td>1004</td>
<td>1056</td>
</tr>
<tr>
<td>6</td>
<td>1026</td>
<td>1015</td>
<td>931</td>
<td>989</td>
<td>1014</td>
<td>979</td>
</tr>
</tbody>
</table>

Debugging

The following program in this section does not run properly. Fix all the compilation errors so that the program will compile successfully. Once the program compiles, compare the output with the sample output, and eliminate any logic errors that may exist. The sample output demonstrates what the program’s output should be once the program’s code has been corrected.

```cpp
#include <iostream>
using std::cout;
using std::endl;
#include <iomanip>
using std::setw;
#include <ctime>
const int NUM_GRADES = 10;
const int NUM_SUDENTS = 3;
int findHighest(int);
int findLowest(int *);
void printDatabase(const int [], const char [], int);
int main()
{
    int student1[NUM_GRADES] = { 0 };
    int student2[NUM_GRADES] = { 76, 89, 81, 42, 66, 93, 104, 91, 71, 85, 105 };
    int student3[NUM_GRADES] = { 65, 69, 91, 89, 82, 93, 72, 76, 79, 99 };
    char names[NUM_SUDENTS][20] = { "Bob", "John", "Joe" };
```cpp
int database[ NUM_SUNDENTS ][ NUM_GRADES ];
int i = 0;
srand( time( 0 ) );

// initialize student1
for ( i = 0; i < NUM_GRADES; i++ )
    student1[ i ] = rand() % 50 + 50;

// initialize database
for ( i = 1; i < NUM_GRADES; i++ ) {
    database[ 0 ][ i ] = student1[ i ];
    database[ 1 ][ i ] = student2[ i ];
    database[ 2 ][ i ] = student3[ i ];
}

// end for
printDatabase( database, studentNames );
for ( i = 0; i < NUM_SUNDENTS; i++ ) {
    cout << studentNames[ i ] << "'s highest grade is: "
        << findHighest( student1 ) << endl
    << studentNames[ i ] << "'s lowest grade is: "
        << findLowest( database[ i ] ) << endl;
}

return 0;

// determine largest grade
int findHighest( int )
{
    int highest = a[ 0 ];
    for ( int i = 1; i <= NUM_GRADES; i++ )
        if ( a[ i ] > highest )
            highest = a[ i ];

    return highest;
}

// determine lowest grade
int findLowest( int a[] )
{
    int lowest = a[ 0 ];
    for ( int i = 1; i < NUM_GRADES; i++ )
        if ( a[ i ] < lowest )
            lowest = a[ i ];

    return lowest;
}
```

// output data
void printDatabase( int a[NUM_GRADES][], char names[][20] )
    cout << "Here is the grade database\n\n"
    << setw(10) << "Name";
for ( int n = 1; n <= NUM_GRADES; n++ )
    cout << setw(4) << n;
cout << endl;
for ( int i = 0; i < NUM_SUDENTS; i++ ) {
    cout << setw(10) << names[i];
    for ( int j = 0; j < NUM_GRADES; j++ )
        cout << setw(4) << a[i][j];
    cout << endl;
} // end for
cout << endl;
} // end printDatabase

Sample Output

Here is the grade database

<table>
<thead>
<tr>
<th>Name</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>56</td>
<td>67</td>
<td>83</td>
<td>81</td>
<td>70</td>
<td>84</td>
<td>94</td>
<td>64</td>
<td>68</td>
<td>86</td>
</tr>
<tr>
<td>John</td>
<td>76</td>
<td>89</td>
<td>81</td>
<td>42</td>
<td>66</td>
<td>93</td>
<td>104</td>
<td>91</td>
<td>71</td>
<td>85</td>
</tr>
<tr>
<td>Joe</td>
<td>65</td>
<td>69</td>
<td>91</td>
<td>89</td>
<td>82</td>
<td>93</td>
<td>72</td>
<td>76</td>
<td>79</td>
<td>99</td>
</tr>
</tbody>
</table>

Bob’s highest grade is: 94
Bob’s lowest grade is: 56
John’s highest grade is: 104
John’s lowest grade is: 42
Joe’s highest grade is: 99
Joe’s lowest grade is: 65

Homework

1. Use a one-dimensional array to solve the following problem. A company pays its salespeople on a commission basis. The salespeople each receive $200 per week plus 9 percent of their gross sales for that week. For example, a salesperson who grosses $5000 in sales in a week receives $200 plus 9 percent of $5000, or a total of $650. Write a program (using an array of counters) that determines how many of the salespeople earned salaries in each of the following ranges (assume that each salesperson’s salary is truncated to an integer amount):
   a) $200–299
   b) $300–399
   c) $400–499
   d) $500–599
   e) $600–699
   f) $700–799
   g) $800–899
h) $900–999
i) $1000 and over

Hints:
- Calculate salary as a double. Then use `static_cast<int>` to truncate the salaries and convert them to integers. Divide by 100 to obtain an array index.
- Make use of the `?:` operator to index the array. (What happens if the index is >= 10?)

Sample output:

```
Enter employee gross sales (-1 to end): 10000  
Employee Commission is $1100.00

Enter employee gross sales (-1 to end): 4235  
Employee Commission is $581.15

Enter employee gross sales (-1 to end): 600  
Employee Commission is $254.00

Enter employee gross sales (-1 to end): 12500  
Employee Commission is $1325.00

Enter employee gross sales (-1 to end): -1
Employees in the range:
$200-$299 : 1
$300-$399 : 0
$400-$499 : 0
$500-$599 : 1
$600-$699 : 0
$700-$799 : 0
$800-$899 : 0
$900-$999 : 0
Over $1000: 2
```

2. Use a one-dimensional array to solve the following problem: Read in 20 numbers, each of which is between 10 and 100, inclusive. As each number is read, validate it and store it in the array only if it is not a duplicate of a number already read. After reading all the values, display only the unique values that the user entered. Provide for the “worst case” in which all 20 numbers are different. Use the smallest possible array to solve this problem.

Hints:
- Compare every value input to all existing array elements. If it is a duplicate, set a flag variable to 1. This flag should be used to determine whether it is necessary to print the value. Use a counter variable to keep track of the number of elements entered into the array and the array position where the next value should be stored.

Sample output:
Lab7 Array II

Objectives
This lab was designed to reinforce programming concepts from Chapter 7 of C++ How To Program: Sixth Edition. In this lab, you will practice:
- Sorting data using the bubble sort algorithm.
The follow-up question and activity also will give you practice:
- Optimizing a program to be more efficient.

Experiments
In the bubble sort algorithm, smaller values gradually “bubble” their way upward to the top of the array like air bubbles rising in water, while the larger values sink to the bottom. The bubble sort makes several passes through the array. On each pass, successive pairs of elements are compared. If a pair is in increasing order (or the values are identical), we leave the values as they are. If a pair is in decreasing order, their values are swapped in the array.
Write a program that sorts an array of 10 integers using bubble sort.

```
Data items in original order
2 6 4 8 10 12 89 68 45 37
Data items in ascending order
2 4 6 8 10 12 37 45 68 89
```

Template
```
// Lab 2: bubblesort.cpp
// This program sorts an array's values into ascending order.
#include <iostream>
```
using std::cout;
using std::endl;

#include <iomanip>
using std::setw;

int main()
{
    const int arraySize = 10; // size of array a
    int a[ arraySize ] = { 2, 6, 4, 8, 10, 12, 89, 68, 45, 37 };
    int hold; // temporary location used to swap array elements
    cout << "Data items in original order\n";
    // output original array
    for ( int i = 0; i < arraySize; i++ )
        cout << setw( 4 ) << a[ i ];
    // bubble sort
    // loop to control number of passes
    /* Write a for header to loop for one iteration less than the size of the array */
    {
        // loop to control number of comparisons per pass
        /* Write a for header to iterate j from 0 and keep looping while j is less than arraySize - 1 */
        {
            // compare side-by-side elements and swap them if
            // first element is greater than second element
            /* Write an if statement to test if element j is greater than element j + 1 */
            {
                /* Write code to swap the values in element j and element j + 1, using hold as temporary storage */
                } // end if
            } // end for
        } // end for
    cout << "\nData items in ascending order\n";
    // output sorted array
    for ( int k = 0; k < arraySize; k++ )
        cout << setw( 4 ) << a[ k ];
    cout << endl;
    return 0; // indicates successful termination
} // end main

Fig. L 7.3 | bubblesort.cpp.

Problem-Solving Tips

1. Each “bubbling” pass through the array brings one element, the
up to its correct position. This means that the program will require arraySize - 1 passes through the array to sort the entire array.

2. Each bubbling pass will look at each pair of adjacent elements and swap them if they are not already in sorted order.

3. To swap two elements, the value of one element will have to be stored in a temporary storage variable while the value of the other element is placed in the first, and then the second element can be replaced with the temporary storage value.

Follow-Up Question and Activity

1. This bubble sort algorithm is inefficient for large arrays. Make the following simple modifications to improve the performance of the bubble sort:
   a) After the first pass, the largest number is guaranteed to be in the highest-numbered element of the array; after the second pass, the two highest numbers are “in place,” and so on. Instead of making nine comparisons on every pass, modify the bubble sort to make eight comparisons on the second pass, seven on the third pass, and so on.
   b) The data in the array may already be in the proper order or near-proper order, so why make nine passes if fewer will suffice? Modify the sort to check at the end of each pass if any swaps have been made. If none have been made, then the data must already be in the proper order, so the program should terminate. If swaps have been made, then at least one more pass is needed.

Program Challenge

A prime integer is any integer greater than 1 that is evenly divisible only by itself and 1. The Sieve of Eratosthenes is a method of finding prime numbers. It operates as follows:

1. Create an array with all elements initialized to 1 (true). Array elements with prime subscripts will remain. All other array elements will eventually be set to zero. You will ignore elements 0 and 1 in this exercise.

2. Starting with array subscript 2, every time an array element is found whose value is 1, iterate through the remainder of the array and set to zero every element whose subscript is a multiple of the subscript for the element with value 1. For array subscript 2, all elements beyond 2 in the array that are multiples of 2 will be set to zero (subscripts 4, 6, 8, 10, etc.); for array subscript 3, all elements beyond 3 in the array that are multiples of 3 will be set to zero (subscripts 6, 9, 12, 15, etc.); and so on.

When this process is complete, the array elements that are still set to one indicate that the subscript is a prime number. These subscripts can then be printed. Write a program that uses an array of
1000 elements to determine and print the prime numbers between 2 and 999. Ignore elements 0 and 1 of the array.

**Homework:**

7.32, 7.37

---

**Lab8 Pointers and Pointer-based Strings**

**Objectives**

- To understand the concept of pointer, master pointer-based indirect organization method of data.
- To master the basic of declare and use pointer in C++, the basic operation of pointer.
- To master the relationship between pointer and array, the pointer-based array operation.
- To master the relationship between pointer and string, the pointer-based string operation.
- To master the use of standard string library function.
- To master the relationship between pointer and function, the pointer-based function operation.

**Experiments**

1. To run the following program and analysis its result.

```cpp
#include <iostream>
using std::cout;
using std::endl;

int main()
{
    int a[] = {1, 3, 5, 7, 9};
    int *p[] = {a, a+1, a+2, a+3, a+4};
    int *p1 = a;

    cout << "Test 1:-------------------------------" << endl;

    cout << "Test 2:-------------------------------" << endl;
    cout << *(++p1) << "\t" << *(p1+2) << "\t" << *(+++p1) << "\t" << *(++p1=88) << "\t" << endl;

    cout << "Test 3:-------------------------------" << endl;
    cout << sizeof(p1) << "\t" << sizeof(*p1) << "\t" << &p1 << "\t" << &*(p1) << "\t" << endl;
}
```
2. To declare an array and input 10 integers into it. Then sort and output the array by pointer-based operation.

3. To declare a function to swap two integer as following every prototype, and analysis carefully their executing.

```c
void swap(int a, int b); // declare a local identifier temp
void swap(int a, int b); // declare a local identifier temp
void swap(int* a, int* b); // declare a local identifier temp
void swap(int* a, int* b); // declare a local identifier temp
void swap(int& a, int& b); // declare a local identifier temp
```

4. To declare a function to get the sum of two integer as following every prototype, and analysis carefully their executing.

```c
int getsum(int a, int b); // declare a local identifier temp, return temp
int* swap(int a, int b); // declare a local identifier temp, return the address of temp
void swap(int a, int b); // declare a local identifier temp*, return temp
void swap(int* a, int* b); // declare a local identifier temp*, return the content of temp
void swap(int* a, int* b); // declare a local identifier temp&, return temp
```

5. To write a function `strcpy(char*s1, const char* s2)` that copies string `s2` into the character array `s1`. (see Fig8.30 in p341)

6. To write a function `strcat(char*s1, const char* s2)` that appends the string `s2` to `s1`. (see Fig8.30 in p341)

7*. At first, we declare a 3-by-5 two-dimensional array. Then transpose it and output result by pointer-based operation.

**Homework:**

Ex8.36, Ex8.38*, Ex8.46