- The product
- The average

P2.5 Write a program that prompts the user for two integers and then prints

- The distance (absolute value of the difference)
- The maximum (the larger of the two)
- The minimum (the smaller of the two)

Hint: The `max` and `min` functions are defined in the `<algorithm>` header.

P2.6 Write a program that prompts the user for a measurement in meters and then converts it to miles, feet, and inches.

P2.7 Write a program that prompts the user for a radius and then prints

- The area and circumference of a circle with that radius
- The volume and surface area of a sphere with that radius

P2.8 Write a program that asks the user for the lengths of the sides of a rectangle and then prints

- The area and perimeter of the rectangle
- The length of the diagonal (use the Pythagorean theorem)

P2.9 Improve the program discussed in the How To 2.1 to allow input of quarters in addition to bills.

P2.10 Write a program that helps a person decide whether to buy a hybrid car. Your program's inputs should be:

- The cost of a new car
- The estimated miles driven per year
- The estimated gas price
- The estimated resale value after 5 years

Compute the total cost of owning the car for 5 years. (For simplicity, we will not take the cost of financing into account.) Obtain realistic prices for a new and used hybrid and a comparable car from the Web. Run your program twice, using today's gas price and 15,000 miles per year. Include pseudocode and the program runs with your assignment.

P2.11 The following pseudocode describes how a bookstore computes the price of an order from the total price and the number of the books that were ordered.

```
Read the total book price and the number of books.
Compute the tax (7.5% of the total book price).
Compute the shipping charge ($2 per book).
The price of the order is the sum of the total book price, the tax, and the shipping charge.
Print the price of the order.
```

Translate this pseudocode into a C++ program.

P2.12 The following pseudocode describes how to turn a string containing a ten-digit phone number (such as "4155551212") into a more readable string with parentheses and dashes, like this: "(415) 555-1212".

```
Take the substring consisting of the first three characters and surround it with "(" and ")". This is the
   area code.
Concatenate the area code, the substring consisting of the next three characters, a hyphen, and the
   substring consisting of the last four characters. This is the formatted number.
```

Translate this pseudocode into a C++ program that reads a telephone number into a string variable, computes the formatted number, and prints it.

P2.13 The following pseudocode describes how to extract the dollars and cents from a price given as a floating-point value. For example, a price 2.95 yields values 2 and 95 for the dollars and cents.

```
Assign the price to an integer variable dollars.
Multiply the difference price - dollars by 100 and add 0.5.
Assign the result to an integer variable cents.
```

Translate this pseudocode into a C++ program. Read a price and print the dollars and cents. Test your program with inputs 2.95 and 4.35.

P2.14 *Giving change.* Implement a program that directs a cashier how to give change. The program has two inputs: the amount due and the amount received from the customer. Display the dollars, quarters, dimes, nickels, and pennies that the customer should receive in return.

P2.15 Write a program that asks the user to input

- The number of gallons of gas in the tank
- The fuel efficiency in miles per gallon
- The price of gas per gallon

Then print the cost per 100 miles and how far the car can go with the gas in the tank.

P2.16 *File names and extensions.* Write a program that prompts the user for the drive letter (`C`), the path (`\Windows\System`), the file name (`Readme`), and the extension (`txt`). Then print the complete file name `C:\Windows\System\Readme.txt`. (If you use UNIX or a Macintosh, skip the drive name and use / instead of \ to separate directories.)

P2.17 Write a program that reads a number between 1,000 and 999,999 from the user and prints it *with a comma separating the thousands*. Here is a sample dialog; the user input is in color:

```
Please enter an integer between 1000 and 999999: 23456
23,456
```

P2.18 Write a program that reads a number between 1,000 and 999,999 from the user, where the user enters a comma in the input. Then print the number without a comma. Here is a sample dialog; the user input is in color:

```
Please enter an integer between 1,000 and 999,999: 23,456
23456
```

Hint: Read the input as a string. Measure the length of the string. Suppose it contains n characters. Then extract substrings consisting of the first $n - 4$ characters and the last three characters.

Your program should print the full description of the card. For example,

```
Enter the card notation: QS
Queen of Spades
```

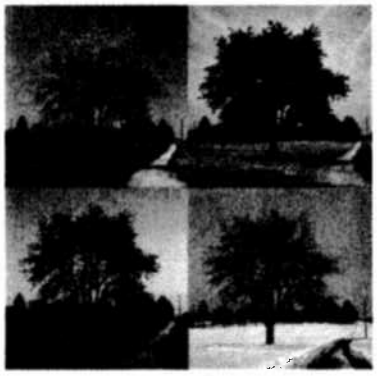
P3.12 When two points in time are compared, each given as hours (in military time, ranging from 0 and 23) and minutes, the following pseudocode determines which comes first.

```
If hour1 < hour2
   time1 comes first.
Else if hour1 and hour2 are the same
   If minute1 < minute2
      time1 comes first.
   Else if minute1 and minute2 are the same
      time1 and time2 are the same.
   Else
      time2 comes first.
Else
   time2 comes first.
```

Write a program that prompts the user for two points in time and prints the time that comes first, then the other time.

P3.13 The following algorithm yields the season (Spring, Summer, Fall, or Winter) for a given month and day.

```
If month is 1, 2, or 3, season = "Winter"
Else if month is 4, 5, or 6, season = "Spring"
Else if month is 7, 8, or 9, season = "Summer"
Else if month is 10, 11, or 12, season = "Fall"
If month is divisible by 3 and day >= 21
   If season is "Winter", season = "Spring"
   Else if season is "Spring", season = "Summer"
   Else if season is "Summer", season = "Fall"
   Else season = "Winter"
```



Write a program that prompts the user for a month and day and then prints the season, as determined by this algorithm.

P3.14 Write a program that reads in two floating-point numbers and tests whether they are the same up to two decimal places. Here are two sample runs.

```
Enter two floating-point numbers: 2.0 1.99998
They are the same up to two decimal places.
Enter two floating-point numbers: 2.0 1.98999
They are different.
```

P3.15 Write a program to simulate a bank transaction. There are two bank accounts: checking and savings. First, ask for the initial balances of the bank accounts; reject negative balances. Then ask for the transactions; options are deposit, withdrawal, and transfer. Then ask for the account; options are checking and savings. Then ask for the amount; reject transactions that overdraw an account. At the end, print the balances of both accounts.

P3.16 Write a program that reads in the name and salary of an employee. Here the salary will denote an *hourly* wage, such as $9.25. Then ask how many hours the employee worked in the past week. Be sure to accept fractional hours. Any overtime work (over 40 hours per week) is paid at 150 percent of the regular wage. Compute the pay. Print a paycheck for the employee.

P3.17 Write a program that prompts for the day and month of the user's birthday and then prints a horoscope. Make up fortunes for programmers, like this:

```
Please enter your birthday (month and day): 6 16
Gemini are experts at figuring out the behavior of complicated programs.
You feel where bugs are coming from and then stay one step ahead. Tonight,
your style wins approval from a tough critic.
```

Each fortune should contain the name of the astrological sign. (You will find the names and date ranges of the signs at a distressingly large number of sites on the Internet.)

P3.18 Write a program that computes taxes for the following schedule:

If your status is Single and if the taxable income is over	but not over	the tax is	of the amount over
$0	$8,000	10%	$0
$8,000	$32,000	$800 + 15%	$8,000
$32,000		$4,400 + 25%	$32,000
If your status is Married and if the taxable income is over	**but not over**	**the tax is**	**of the amount over**
$0	$16,000	10%	$0
$16,000	$64,000	$1,600 + 15%	$16,000
$64,000		$8,800 + 25%	$64,000

P3.19 The original U.S. income tax of 1913 was quite simple. The tax was

- 1 percent on the first $50,000.
- 2 percent on the amount over $50,000 up to $75,000.
- 3 percent on the amount over $75,000 up to $100,000.
- 4 percent on the amount over $100,000 up to $250,000.
- 5 percent on the amount over $250,000 up to $500,000.
- 6 percent on the amount over $500,000.

There was no separate schedule for single or married taxpayers. Write a program that computes the income tax according to this schedule.

P3.20 The `tax.cpp` program uses a simplified version of the 2008 U.S. income tax schedule. Look up the tax brackets and rates for the current year, for both single and married filers, and implement a program that computes the actual income tax.

Your task is to simulate a portion of the control software for the vehicle. The input is a sequence of values for the switches and the gear shift, in the following order:

- Dashboard switches for left and right sliding door, child lock, and master unlock (0 for off or 1 for activated)
- Inside and outside handles on the left and right sliding doors (0 or 1)
- The gear shift setting (one of P N D 1 2 3 R).

A typical input would be `0 0 0 1 0 1 0 0 P`.

Print "left door opens" and/or "right door opens" as appropriate. If neither door opens, print "both doors stay closed".

Engineering P3.31 Sound level L in units of decibel (dB) is determined by

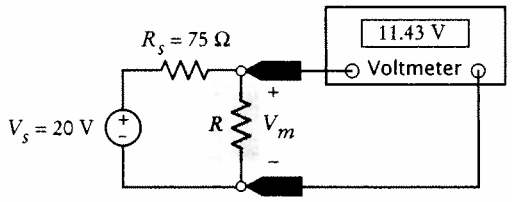
$$L = 20 \log_{10}(p/p_0)$$

where p is the sound pressure of the sound (in Pascals, abbreviated Pa), and p_0 is a reference sound pressure equal to 20×10^{-6} Pa (where L is 0 dB). The following table gives descriptions for certain sound levels.

Threshold of pain	130 dB
Possible hearing damage	120 dB
Jack hammer at 1 m	100 dB
Traffic on a busy roadway at 10 m	90 dB
Normal conversation	60 dB
Calm library	30 dB
Light leaf rustling	0 dB

Write a program that reads a value and a unit, either dB or Pa, and then prints the closest description from the list above.

Engineering P3.32 The electric circuit shown below is designed to measure the temperature of the gas in a chamber.



The resistor R represents a temperature sensor enclosed in the chamber. The resistance R, in Ω, is related to the temperature T, in °C, by the equation

$$R = R_0 + kT$$

In this device, assume $R_0 = 100\ \Omega$ and $k = 0.5$. The voltmeter displays the value of the voltage, V_m, across the sensor. This voltage V_m indicates the temperature, T, of the gas according to the equation

$$T = \frac{R}{k} - \frac{R_0}{k} = \frac{R_s}{k}\frac{V_m}{V_s - V_m} - \frac{R_0}{k}$$

P4.15 Write a program that reads a word and prints all substrings, sorted by length. For example, if the user provides the input `"rum"`, the program prints

```
r
u
m
ru
um
rum
```

P4.16 Write a program that reads a number and prints all of its *binary digits:* Print the remainder `number % 2`, then replace the number with `number / 2`. Keep going until the number is 0. For example, if the user provides the input 13, the output should be

```
1
0
1
1
```

P4.17 *Mean and standard deviation.* Write a program that reads a set of floating-point data values. Choose an appropriate mechanism for prompting for the end of the data set. When all values have been read, print out the count of the values, the average, and the standard deviation. The average of a data set $\{x_1, \ldots, x_n\}$ is $\bar{x} = \Sigma x_i / n$, where $\Sigma x_i = x_1 + \ldots + x_n$ is the sum of the input values. The standard deviation is

$$s = \sqrt{\frac{\sum (x_i - \bar{x})^2}{n - 1}}$$

However, this formula is not suitable for the task. By the time the program has computed $\bar{x}$, the individual x_i are long gone. Until you know how to save these values, use the numerically less stable formula

$$s = \sqrt{\frac{\sum x_i^2 - \frac{1}{n}\left(\sum x_i\right)^2}{n - 1}}$$

You can compute this quantity by keeping track of the count, the sum, and the sum of squares as you process the input values.

P4.18 The *Fibonacci numbers* are defined by the sequence

$$f_1 = 1$$
$$f_2 = 1$$
$$f_n = f_{n-1} + f_{n-2}$$

Fibonacci numbers describe the growth of a rabbit population.

Reformulate that as

```
fold1 = 1;
fold2 = 1;
fnew = fold1 + fold2;
```

After that, discard `fold2`, which is no longer needed, and set `fold2` to `fold1` and `fold1` to `fnew`. Repeat `fnew` an appropriate number of times.

Implement a program that computes the Fibonacci numbers in that way.

P4.19 *Factoring of integers.* Write a program that asks the user for an integer and then prints out all its factors. For example, when the user enters 150, the program should print

```
2
3
5
5
```

P4.20 *Prime numbers.* Write a program that prompts the user for an integer and then prints out all prime numbers up to that integer. For example, when the user enters 20, the program should print

```
2
3
5
7
11
13
17
19
```

Recall that a number is a prime number if it is not divisible by any number except 1 and itself.

P4.21 Write a program that prints a multiplication table, like this:

```
 1   2   3   4   5   6   7   8   9  10
 2   4   6   8  10  12  14  16  18  20
 3   6   9  12  15  18  21  24  27  30
...
10  20  30  40  50  60  70  80  90 100
```

P4.22 Write a program that reads an integer and displays, using asterisks, a filled and hollow square, placed next to each other. For example if the side length is 5, the program should display

```
***** *****
***** *   *
***** *   *
***** *   *
***** *****
```

P4.23 Write a program that reads an integer and displays, using asterisks, a filled diamond of the given side length. For example, if the side length is 4, the program should display

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

P4.24 *The game of Nim.* This is a well-known game with a number of variants. The following variant has an interesting winning strategy. Two players alternately take marbles from a pile. In each move, a player chooses how many marbles to take. The player must take at least one but at most half of the marbles. Then the other player takes a turn. The player who takes the last marble loses.