1. Random Credit Card Numbers (10 points)

Write a complete C program that generates 10 random 16-digit credit
card numbers for a Discover card. The first 6 digits of each credit
card number are reserved for the issuer number. For Discover cards
these digits must be "6011-xx". The remaining ten digits are the
account number. Print the credit card numbers using the standard
format: 6011-xxxx-xxxx-xxxx.

The card numbers must be different every time your program is executed.

SAMPLE RUN:

6011-3834-4218-9357
6011-5536-3701-3912
6011-1663-8203-7559
6011-4510-8871-8681
6011-8568-2780-4730
6011-1436-1756-1762
6011-4189-3250-7716
6011-0609-1802-2561
6011-9681-8075-1019
6011-3003-5535-4391

2. Seven Segment Indicators (15 point)

Seven Segment indicators are commonly used to display numbers 0-9.
It is possible, however, to represent some letters as well. For example
the first six letters of the English alphabet that are also used in the
hexadecimal number system can be displayed as:

```
 _       _   _   _
|_| |_  |   _| |_ |_
| | |_| |_ |_| |_ |
```

Write a complete C program that generates two random letters in the
range [A .. F] and prints them on the screen side by side using the
ASCII encoding of their seven-segment indicator representation (as
shown in the sample runs given below). The program must come up with a
different pair of letters every time you run it. It is OK if during
some (but not all) runs the two random letters are the same.

SAMPLE RUN #1:

```
 _   _
|_| |_
| | |
```

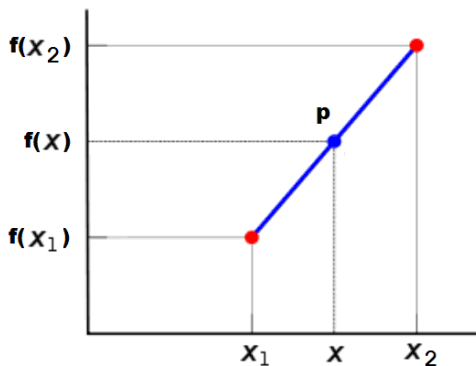SAMPLE RUN #2:

```
 _
|_   _|
|_  |_|
```

SAMPLE RUN #3:

```
 _    _
|    |_
|_   |_|
```

3. Bilinear Interpolation (15 points)

MATH BACKGROUND: Linear Interpolation

Linear interpolation is a common method for estimating the value of a function at a point x for which no data is available, but data exists for points located on either side of x on the horizontal axis. Given two points x1 and x2 and the value of the function evaluated at these points f(x1) and f(x2), using linear interpolation one can estimate the value of the function f(x) at another point x that lies between x1 and x2. The linear interpolation formula is given below.

Linear Interpolation Formula:

$$f(x) = \frac{(x - x1)}{(x2 - x1)} f(x2) \ + \ \frac{(x2 - x)}{(x2 - x1)} f(x1)$$

In other words, the estimation is performed by fitting a straight line between the points (x1, f(x1)) and (x2, f(x2)) and using this line to estimate the values of the function in the interval (x1, x2).
Note that we don't even need to know the analytical form of the function; we only need to know its values at two points.

MATH BACKGROUND: Bilinear Interpolation

Bilinear interpolation is a natural extension of linear interpolation
that can be used with functions of two variables, e.g., f(x,y).
Once again, we don't need to know the analytical form of the function.
We only need to know its value at four points that form a rectangle,
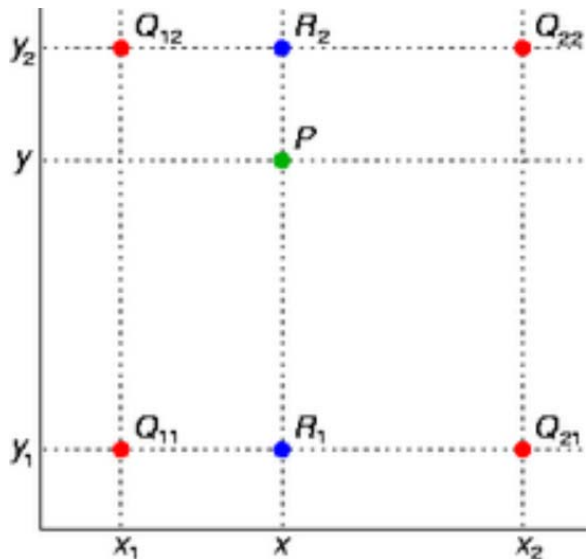which contains the point of interest (x,y).

Image Source: Wikipedia

To find an estimate for the value of the function at the point (x,y),
where x1 < x < x2, and y1 < y < y2, we can perform linear interpolation
twice in the x direction followed by a third linear interpolation in
the y direction.

For example, in the image above, we can interpolate the function
between the points Q11=f(x1,y1) and Q21=f(x2, y1) to find approximation
for its value at the point R1=(x,y1). We can also perform another
linear interpolation between the points Q12=f(x1,y2) and Q22=f(x2,y2)
to find an interpolated value of the function at the point R2=(x,y2).
Finally, we can perform a third linear interpolation between R1 and R2
to find an estimate for the value of the function at the point P=(x,y).

PROBLEM DESCRIPTION

Write a complete C program that approximates and prints the value of an
analytically unknown function at the point (x,y) using bilinear
interpolation. The program must prompt the user to enter the following
double numbers:

      * x1, x2, y1, and y2;

      * Q11, Q12, Q21, Q22, which correspond to the value of
        the function at (x1,y1), (x1,y2), (x2,y1), and (x2,y2).

      * x, y -- the coordinates of the point for at which we
            would like to approximate the value of the function

```
SAMPLE RUN #1:
==============

Please enter x1:     12
Please enter x2:     13
Please enter y1:     18
Please enter y2:     19
Please enter x:      12.6
Please enter y:      18.7
Please enter Q11=f(x1,y1): 135
Please enter Q12=f(x1,y2): 85
Please enter Q21=f(x2,y1): 147
Please enter Q22=f(x2,y2): 215

The interpolated value of the function at R1=(x,y1) is 142.2
The interpolated value of the function at R2=(x,y2) is 163

The interpolated value of the function at (x=12.6, y=18.7) is equal to 156.76




SAMPLE RUN #2:
==============

Please enter x1:     10
Please enter x2:     20
Please enter y1:     40
Please enter y2:     50
Please enter x:      14
Please enter y:      48
Please enter Q11=f(x1,y1): 1.5
Please enter Q12=f(x1,y2): 3.3
Please enter Q21=f(x2,y1): 2.1
Please enter Q22=f(x2,y2): 4.5

The interpolated value of the function at R1=(x,y1) is 1.74
The interpolated value of the function at R2=(x,y2) is 3.78

The interpolated value of the function at (x=14, y=48) is equal to 3.372
```

4. ASCII to Binary Converter (15 points)

Write a complete C program that takes a character array and prints its
letters in binary format on the screen. You can assume that the
character array is hardcoded in the program. In other words, you don't
have to read it from the keyboard. The following will suffice:

```
char text[] = "Halloween";
//char text[] = "Secret Message";
```

To test the program you can keep several text messages (commented out)
and uncomment only one for testing.

The program must automatically figure out the length of the message
by scanning the array until it finds a character with an ASCII value of
0 (i.e., the standard string termination character '\0').

Each letter of the message must be printed on a single line
as an 8-bit binary number that corresponds to the ASCII code
of the letter (see the sample runs below).

The program must define and call a function that does the ASCII to
binary conversion for each letter and also prints the encoded letter.
The termination character should not be printed as part of the message.

Sample Run #1:

```
Halloween
01001000
01100001
01101100
01101100
01101111
01110111
01100101
01100101
01101110
```

Sample Run #2:

```
Secret Message
01010011
01100101
01100011
01110010
01100101
01110100
00100000
01001101
01100101
01110011
01110011
01100001
01100111
01100101
```

5. Matching Numbers (15 points)

Write a complete C program that asks the user to enter a number N in the range 1 to 20 (inclusive). If the user enters a number outside this range the program must ask the user to enter the number again. The program must then generate two sets of N random numbers, where each number is in the range [0,..,9], print the two sets on the screen (one set per line), and then print the numbers that appear in both sets, if any. Format the output similar to the sample runs shown below.

```
SAMPLE RUN #1:
==============

Please enter N in the range [1,20]: 7

First set:   0, 5, 2, 5, 2, 6, 8,
Second set:  3, 1, 8, 9, 3, 5, 3,

The following numbers appear in both sets:
 5, 8,


SAMPLE RUN #2:
==============

Please enter N in the range [1,20]: 32
Please enter N in the range [1,20]: -1
Please enter N in the range [1,20]: 2

First set:   5, 3,
Second set:  8, 9,

There are no common numbers between the two sets.


SAMPLE RUN #3:
==============

Please enter N in the range [1,20]: 20

First set:   4, 0, 8, 1, 5, 5, 0, 1, 3, 3, 9, 4, 0, 2, 6, 0, 9, 4, 1, 0,
Second set:  9, 8, 6, 2, 8, 9, 9, 3, 4, 9, 7, 4, 7, 9, 5, 5, 2, 5, 6, 2,

The following numbers appear in both sets:
 2, 3, 4, 5, 6, 8, 9,
```