

# Computer Programming

## C Programming Assignment #2 2001/2

### **Brief**

Write a program that generates sets of 6 lotto numbers and displays them on-screen.

### **Overview of Program Flow**

When the program is run the user should be asked how much money they wish to spend on the Lotto. The user will type the amount of money (in pounds) and the program should generate the appropriate number of sets of 6 numbers (6 numbers for 75p).

When the program has displayed the sets of numbers on the screen the user should then be asked if they wish to generate more sets of numbers. If the user declines then the program finishes. (This test may alternatively be carried out prior to generating any numbers).

### **Restrictions**

1. Negative monetary values are not accepted and must be handled in some way.
2. A minimum of two plays must be enforced. (A minimum play is £1.50.)
3. Additional plays are £0.75 each.
4. Numbers generated must be in the range of 1-42.

### **Assumptions**

1. No allowance for duplicate numbers need be made, but if you successfully cater for this extra marks will be awarded.
2. No currency symbols need be input, but you may wish to output some.

### **Suggestions**

1. The program will need to use *loops* to repeatedly create random numbers, as well as to repeatedly create sets of six numbers, as well as to ask if the user wishes to re-run the program. The sixth chapter of the course text book deals with loops. While you should work through the whole chapter and try all the exercises, only the *while* loop construct need be used for this assignment. You may wish to use either the *do-while* construct or the *for* construct, but they are not necessary.

2. Random numbers are generated using a combination of the *srand (n)* and the *rand ()* functions. Note that there will be two extra *#include* directives, *#include <stdlib.h>* and *#include <time.h>* which will make the required functions available.

a) The *srand (n)* function is used only once, at the start of the program to 'seed' the random number generator with the integer value n. As starting the program with the same value of n each time will always give the same values of 'random' numbers, we normally seed the generator with the time as the value for n. The time can be returned with the *time(NULL)* function. Refer to page 119 of the text book for an explanation of the *srand*, *rand* and *time* functions. Refer to and experiment with the program on page 120 (type it in!) to get experience with using these functions.

b) Use the command:

**OneNumber = rand ()%42**

in your program to generate a value between 0 and 42 inclusive and to assign it into the variable OneNumber (which you would declare).

3. You will need to use more than one loop. Loops which are positioned within other loops are called *nested* loops. Pages 51 and 52 has more details. If you draw a suitable flow chart the nesting of loops is easier to cope with.

4. When drawing your flow chart, it might be easier on this occasion to work from the middle out! At the heart of your program you'll have a slightly unusual arrangement of boxes at the centre of the assignment, such as those shown [here](#). This represents a *while* loop. (It's not necessary to put the word while in the diamond - that's just for your benefit).

### ***Test Data***

<b>Input Data</b>	<b>Expected Result</b>
-3	Error message generated
1.50	2 sets of 6 numbers
3.00	4 sets of 6 numbers
0	No numbers generated
3.75	5 sets of 6 numbers

### ***Sample Output***

This some [sample program output](#). Note that this will not be *required* output but *sample* output.

### ***Submission Criteria***

Provide evidence of your planning process, the commented source code and sample data used. You should also include a hand-drawn as well as a SmartDraw formatted flow-chart. Any other appropriate documentation may be included if you wish. The submission should be preceded with a signed copy of the '[my own work](#)' form.

Submission materials should be submitted in paper form to the submission box in Room 15.

**NOTE:** There is a lot of deliberate lee-way given in this assignment specification. You are advised to exploit this to the full.

**Due date: December-2001**