# COMP2611: Computer Organization

## MIPS programming

## Overview

- You will learn the following in this lab:
  - ❑ MIPS programming by practicing it in MARS.

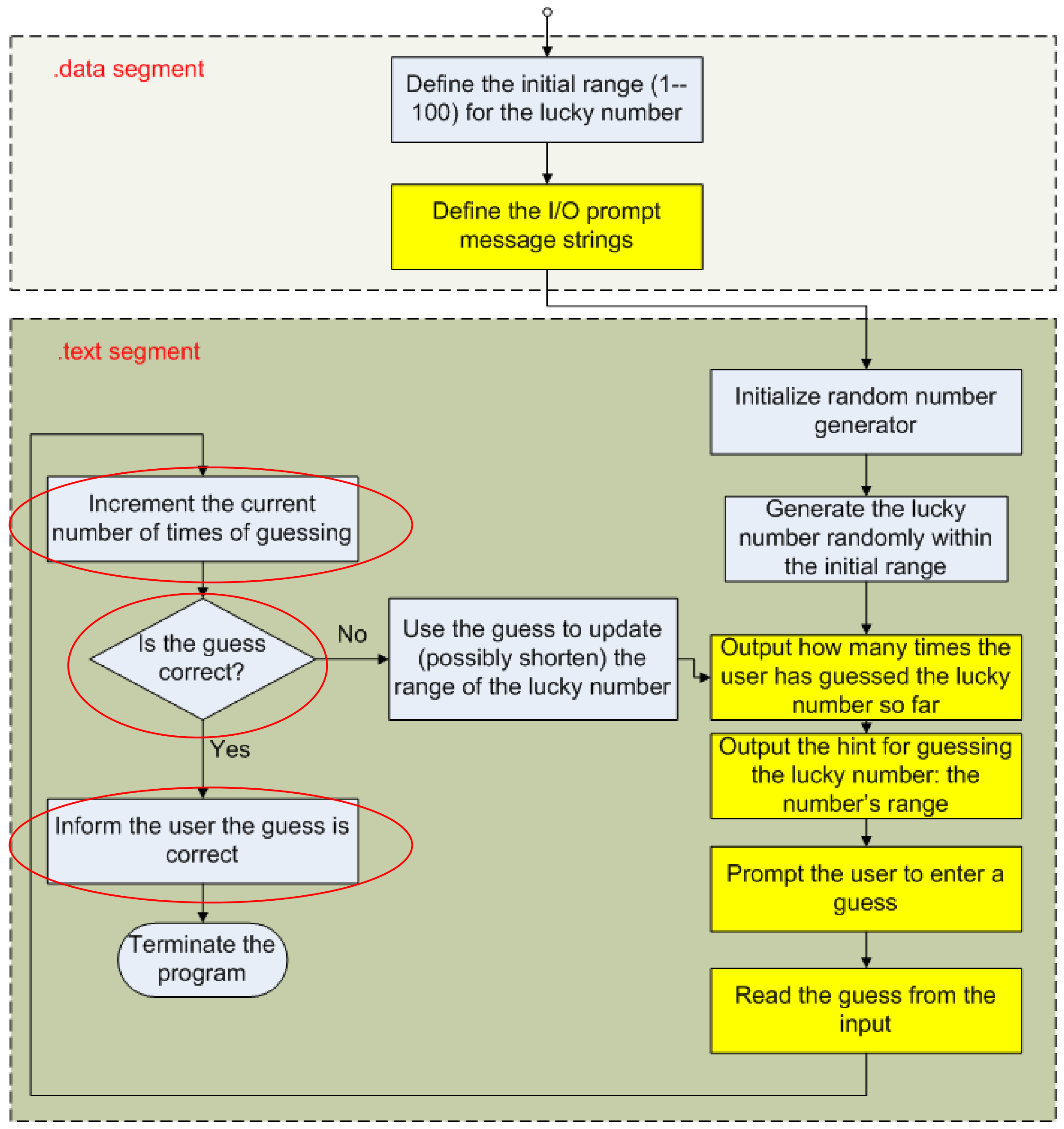
# The number guessing game

- Complete the skeleton program code of guessingGame.s by following the comments on the code. You may also see the logic flow in the file guessingGame.cpp
  - The program picks a lucky number randomly from the interval 1 to 100. It then prompts the user to guess the number by telling him/her this interval as a hint.
  - If the input guess is correct, the program tells the user and then terminates.
  - Otherwise, the user's guess will be used to update (shorten) the interval as the hint for the user's next guess (until a guess is correct).
  - The number of guesses the user has made so far should also be displayed (until a correct guess is made).
  - The program code for updating the number interval as the hint has already been completed. You just need to complete the other syscall-related codes (as suggested by the code comments), e.g., I/O syscalls.

## The number guessing game

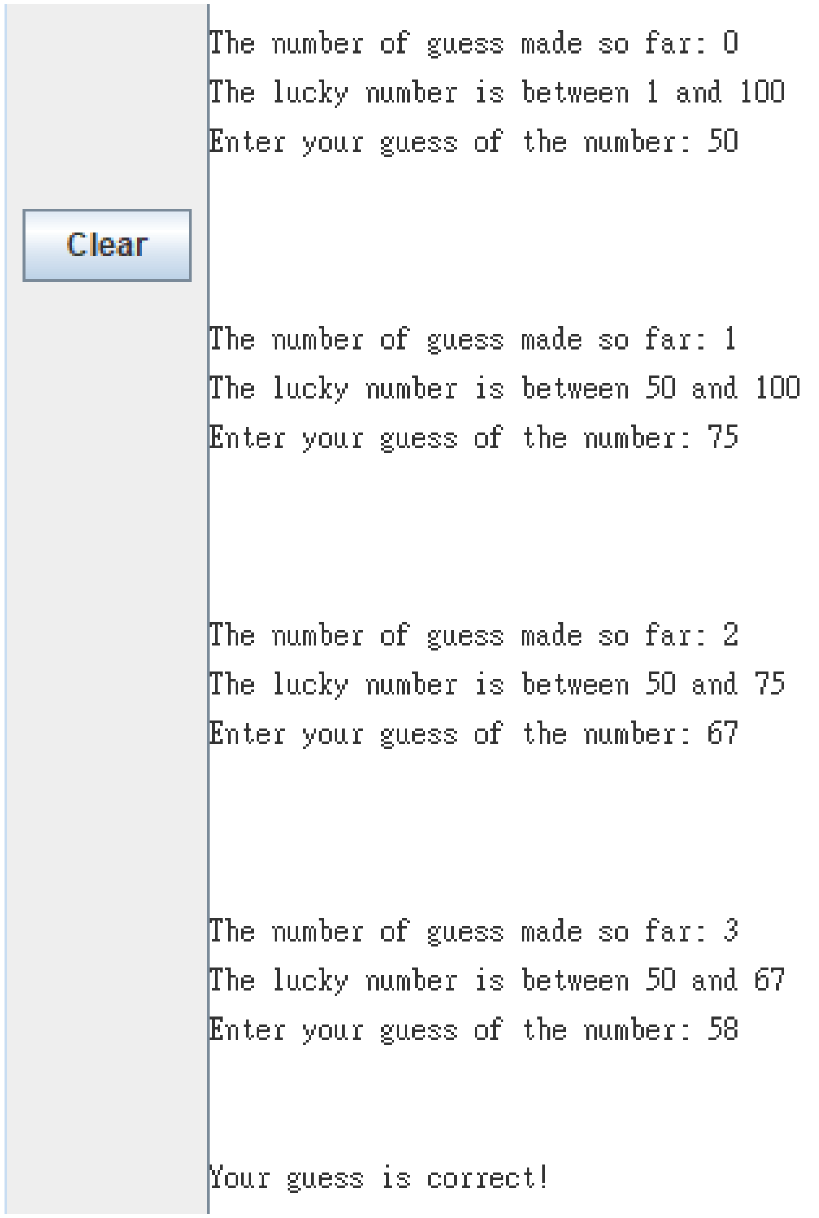
- Follow the comments on the code to complete the program. The program tasks to be completed are also highlighted in yellow or circle in Red in the flow chart on the next page.

# Program flow



.data segment
Define the initial range (1--100) for the lucky number
Define the I/O prompt message strings
.text segment
Initialize random number generator
Generate the lucky number randomly within the initial range
Output how many times the user has guessed the lucky number so far
Output the hint for guessing the lucky number: the number's range
Prompt the user to enter a guess
Read the guess from the input
Increment the current number of times of guessing
Is the guess correct?
No
Use the guess to update (possibly shorten) the range of the lucky number
Yes
Inform the user the guess is correct
Terminate the program

# Output Sample

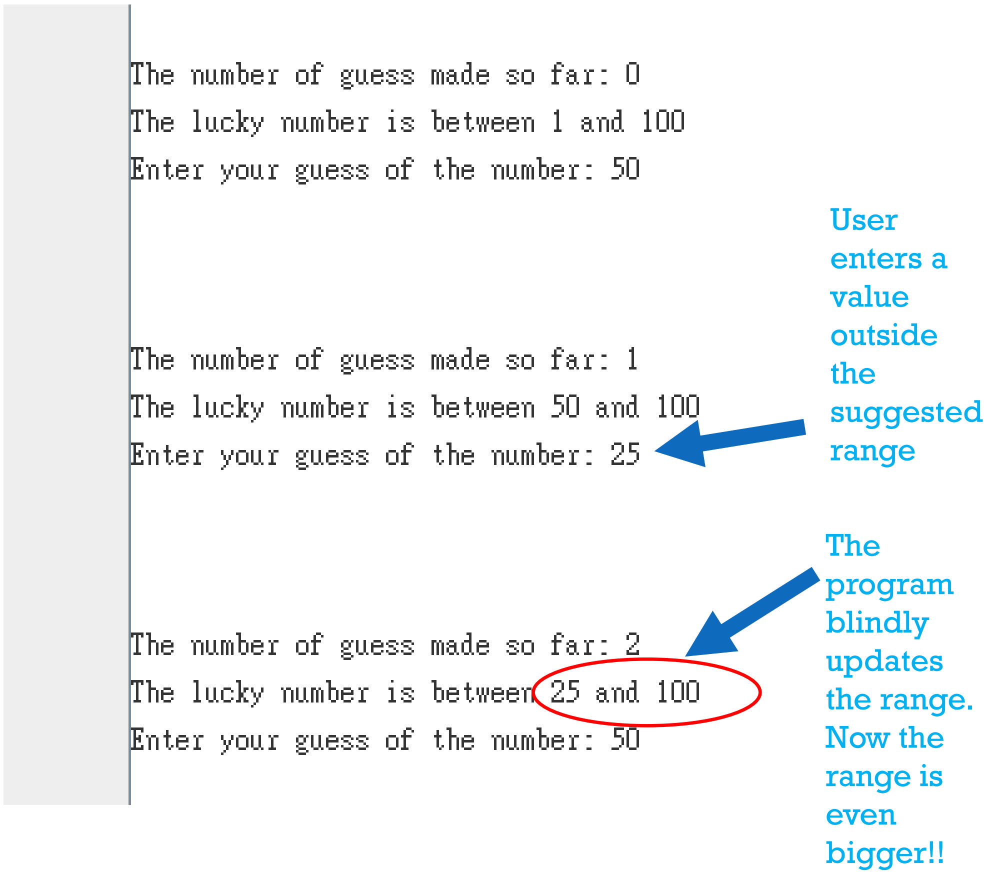


```
The number of guess made so far: 0
The lucky number is between 1 and 100
Enter your guess of the number: 50



The number of guess made so far: 1
The lucky number is between 50 and 100
Enter your guess of the number: 75



The number of guess made so far: 2
The lucky number is between 50 and 75
Enter your guess of the number: 67



The number of guess made so far: 3
The lucky number is between 50 and 67
Enter your guess of the number: 58


Your guess is correct!
```

# Possible enhancement

- The first version we assumed that the user will always enter a number within the range being suggested. If the user enters a number outside the suggested range, no error message will be displayed, instead the program will blindly update the range using the user entered value. See the example below.
  - Prompt an error message to guide the user to guess within the range

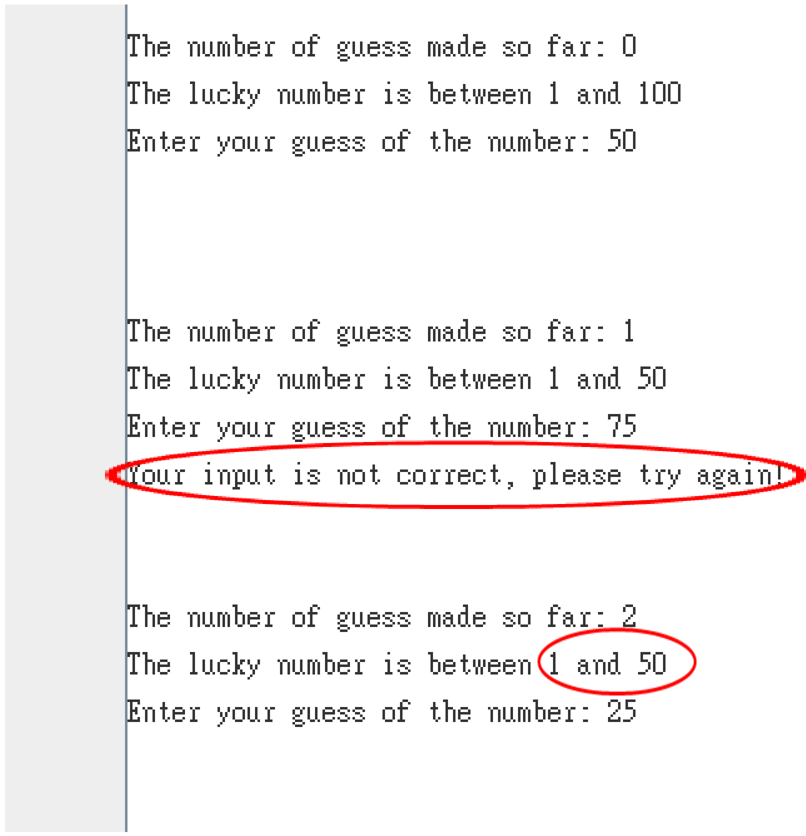
## Version 1



```
The number of guess made so far: 0
The lucky number is between 1 and 100
Enter your guess of the number: 50


The number of guess made so far: 1
The lucky number is between 50 and 100
Enter your guess of the number: 25


The number of guess made so far: 2
The lucky number is between 25 and 100
Enter your guess of the number: 50
```

User enters a value outside the suggested range
The program blindly updates the range. Now the range is even bigger!!

## Version 2



```
The number of guess made so far: 0
The lucky number is between 1 and 100
Enter your guess of the number: 50


The number of guess made so far: 1
The lucky number is between 1 and 50
Enter your guess of the number: 75
Your input is not correct, please try again!


The number of guess made so far: 2
The lucky number is between 1 and 50
Enter your guess of the number: 25
```

## Further improvement

- There is at least two further improvements for the game:
  - ❑Validate the user input if it is between 1 to 100 (both inclusive). For example, if the user inputs 101, a warning message should be outputted.
  - ❑Implement 2 players mode of the game. For example, they will take turn to guess the number, and there will be a message to state which player win the game finally.

## Extra Exercise

- Write MIPS program that:
  - ❑ prompts the user for two integer inputs,
  - ❑ displays the product of the two integers,
  - ❑ terminates using a syscall service after displaying the product.
- You do not need to verify the correctness of the input integers.
- You are not allowed to use mult / multu instructions (hint: using loop)

# Conclusion

- You have learnt:
  - ❑MIPS programming for the number guessing game in MARS.