INTRODUCTION TO PYTHON ALGORITHMS USING PYTHON

CHARIS CODING CLUB

WWW.CHARISCODINGCLUB.COM

COURSE OUTLINE

- LESSON1: INTRODUCTION TO ALGORITHMS, REFRESHER, GIT
- LESSON2: TACKLING ALGORITHMIC PROBLEMS – EUREKA! CHALLENGE
- LESSON3: STRING METHOD AND STRING CHALLENGE
- LESSON4: NEW PYTHON METHODS FOR SCRIPTING
- LESSON5: USING FLOW CHARTS TO DESCRIBE YOUR ALGORITHM
- LESSON6: FUNCTIONS AND CLASSES
- LESSON7: CHALLENGE 5 AND 6
- LESSON8: SIMPLE GAME
- LESSON9: PROJECT PART 1
- LESSON10: PROJECT PART 2

LESSON 5:
USING FLOW CHARTS TO DESCRIBE YOUR ALGORITHM

OBJECTIVES:

TO MAKE US UNDERSTAND WHAT FLOW CHARTS ARE

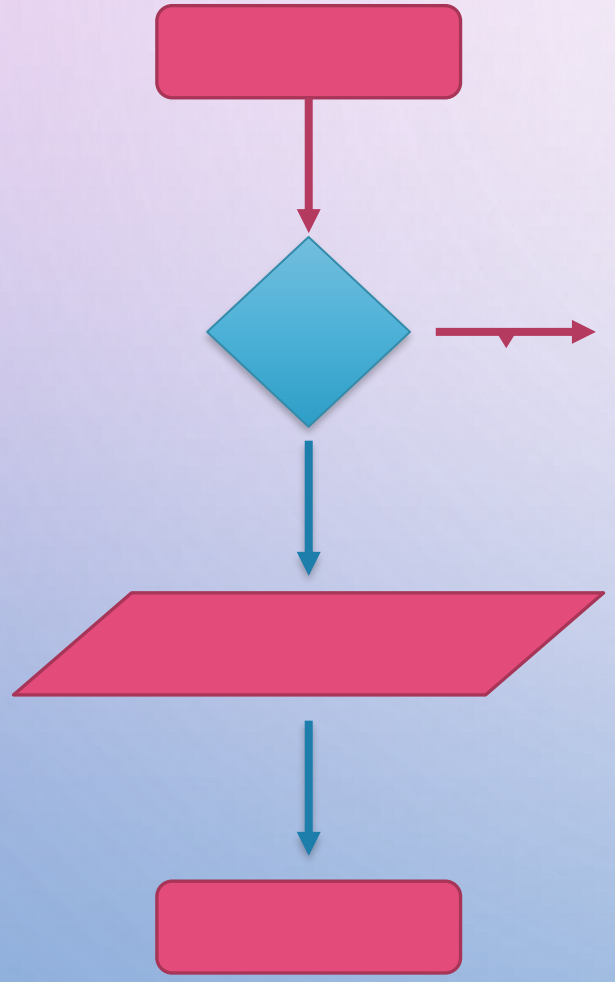
TO BE ABLE TO USE FLOW CHART TO STRUCTURE OUR DESCRIBE OUR SOLUTIONS

TO WRITE CODES FROM GIVEN FLOW CHARTS

RECAP FROM LAST CLASS:

- LET'S SEE OUR SOLUTIONS TO LAST WEEK'S CLASSWORK/ASSIGNMENT.
- SO FAR, WE HAVE LEARNED HOW TO USE NEW METHODS IN TACKLING ALGORITHMIC PROBLEMS.
- REMEMBER TO GO OVER THESE NEW METHODS AND BE SURE THAT YOU'VE FULLY GRASPED THEM.

FLOW CHARTS



Flow charts are a pictoral (using pictures) representation describing an algorithm.

FLOWCHARTS FOR ALGORITHMS

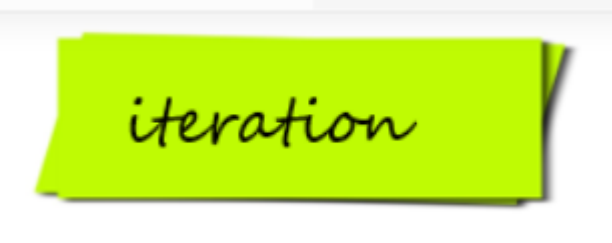
Algorithm is a step-by-step set of instructions that the computer will have to follow to solve a problem or complete a task.

When writing our algorithms, there are three general properties of algorithmic designs which we must look out for. These properties are:

SEQUENCING

Algorithms consist of step by step instructions which are listed in order and will be executed in the sae order, one instruction at a time: this is called sequencing.

ITERATION



On occasions a set of instructions needs to be **repeated several times** which is done in programming using a **loop**: this is called **iteration**.

SELECTION

Computers also have to **take decisions** as to whether or not to run a set of instructions or to bypass these instructions. In programming these decisions are coded using **IF statements**: this is called **selection**.

PSEUDO-CODE

Complex algorithms can use a range of sequencing, iteration and selection blocks.

To design an algorithm you can draw a **flowchart** or write **pseudo-code.**

Your algorithm can then be converted by a programmer using the programming language of their choice (e.g. Python, Java, etc)

EXAMPLE CHALLENGE

HIGHER OR LOWER NUMBER GAME

Challenge Description

For this challenge you will design and write a program to play against the computer. The computer will display a random number between 1 and 100. The user will have to try to guess this number. For each guess the computer will inform the user if the number to guess is higher or lower than the user guess. The program will end once the user guess matches the number to guess.

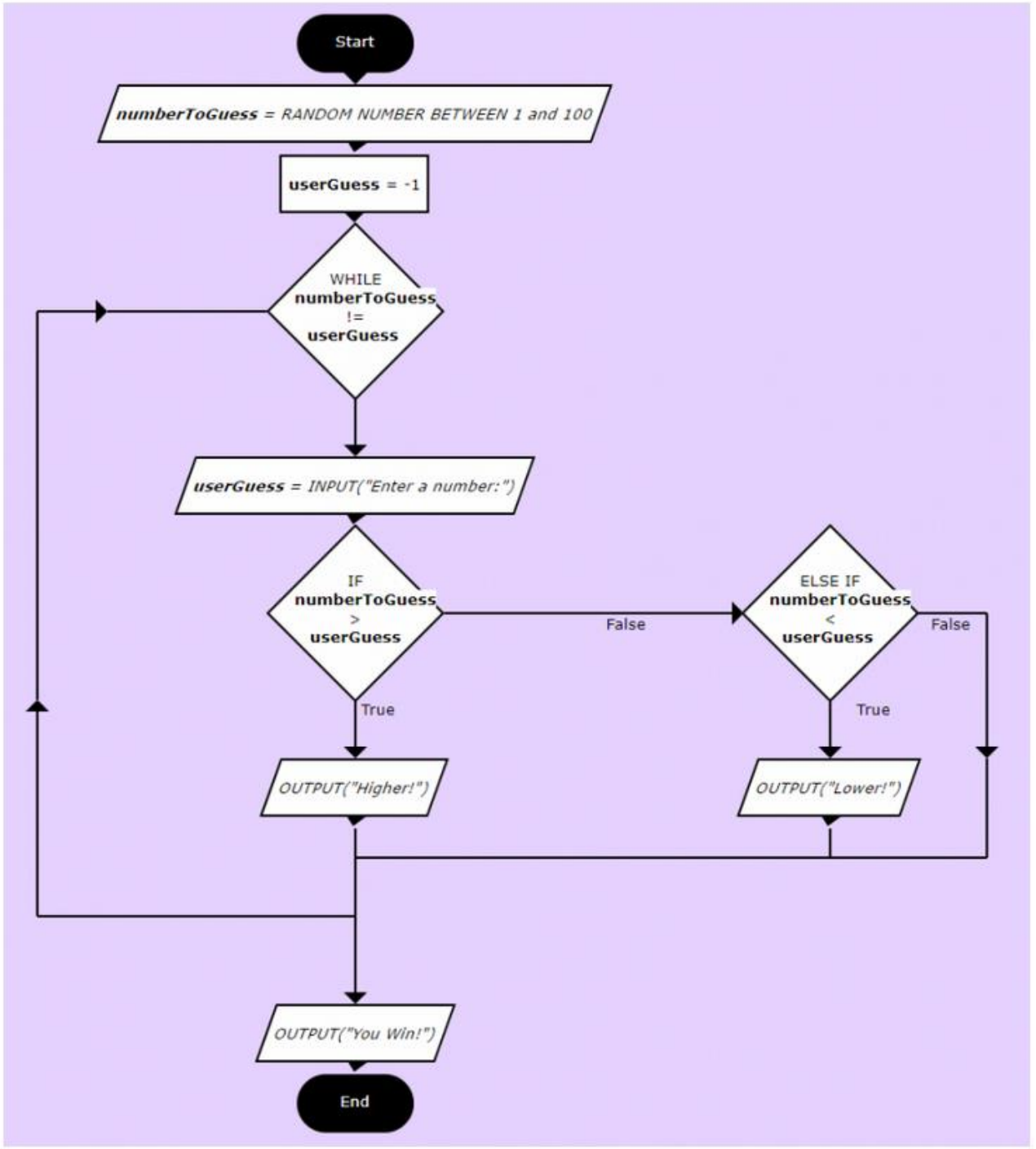
Learning Objectives

By completing this challenge you are going to use selection (IF statements) and iteration (While loop). You will use comparison operators such as > , < and == to compare numbers. You will use variables to store the value of a random number and retrieve user inputs.

TASK 1: FLOWCHART

Use the following link to replicate the flowchart you see below:

https://www.101computing.net/flowchart/


Start
numberToGuess = RANDOM NUMBER BETWEEN 1 and 100
userGuess = -1
WHILE numberToGuess != userGuess
userGuess = INPUT("Enter a number:")
IF numberToGuess > userGuess
False
ELSE IF numberToGuess < userGuess
False
True
True
OUTPUT("Higher!")
OUTPUT("Lower!")
OUTPUT("You Win!")
End

Now, let's attempt to convert the flowchart into code:

Do you want to try it out? Here's a good starting point:

```
# Higher or Lower Number

from random import randint

numberToGuess = randint(1,100)

#Complete the code here...
```

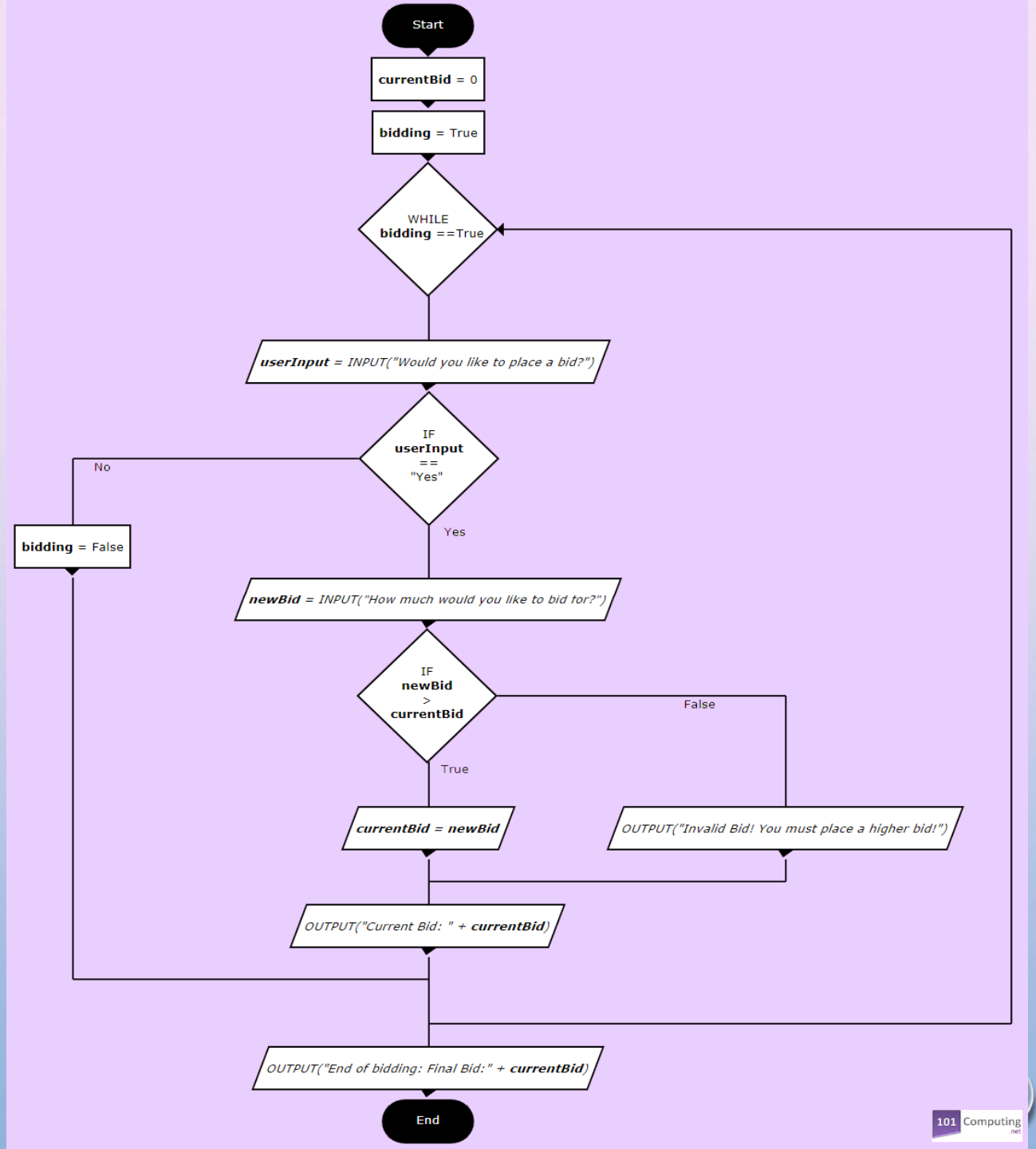
Completed Code:

```
from random import randint

numberToGuess = randint(1,100)

#Complete the code here...

userGuess = -1

while numberToGuess != userGuess:
    userGuess = int(input("Enter a number: "))

    if numberToGuess > userGuess:
        print("Guess higher.")
    elif numberToGuess < userGuess:
        print("Guess lower.")

print("You win!")
```

ASSIGNMENT CHALLENGE:

BIDDING PROCESS FLOWCHART

For this challenge we will consider the bidding process used within online auction websites such as eBay.
Our aim is to create a system that will accept bids from the end-user, check if the bid being placed is greater than the current bid and if so, update the value of the current bid.
Here is the flowchart for our Bidding System:


Start
currentBid = 0
bidding = True
WHILE
bidding ==True
userInput = INPUT("Would you like to place a bid?")
IF
userInput
==
"Yes"
No
Yes
bidding = False
newBid = INPUT("How much would you like to bid for?")
IF
newBid
>
currentBid
False
True
currentBid = newBid
OUTPUT("Invalid Bid! You must place a higher bid!")
OUTPUT("Current Bid: " + currentBid)
OUTPUT("End of bidding: Final Bid:" + currentBid)
End

Make Predictions:

- Check the above flowchart to understand how this algorithm will work. Based on this flowchart can you answer the following questions?

1. What will be the first question asked to the end user?

Make Predictions:

.

2. How does this algorithm decide if a bid is a valid bid or not?

Make Predictions:

.

3. How does this algorithm decide when the bidding process ends?

Make Predictions:

.

4. What will be the last message displayed to the end user?

Write the code

- Use the flowchart above to write the algorithm for the billing system.

```
# Bidding Algorithm from biddding process flow chart

# Complete the code below:

currentBid = 0
```

Make sure you complete your assignment and do it well.

Because next class, we'll be pushing your code to github!

☺