COMP1021
Introduction to Computer Science

Making Decisions

David Rossiter and Gibson Lam

Outcomes

- After completing this presentation, you are expected to be able to:
  1. Use the if statements (if, elif and else) to make decisions in a Python program
  2. Write code using nested if statements

The if Statement

- The if statement is used to decide whether some code will be executed
- Here is a simple example:

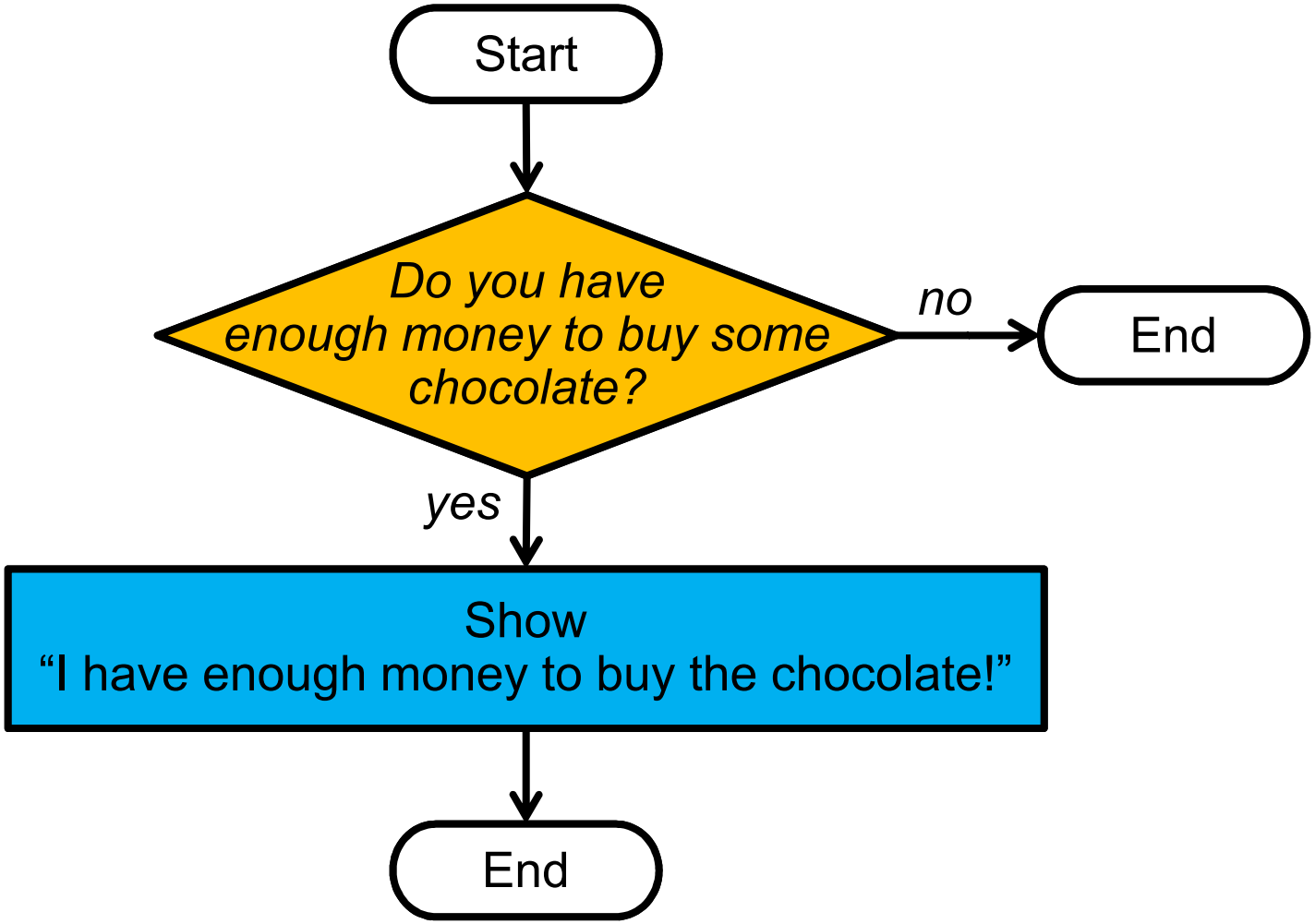
```
cost_of_chocolate = 10
money_in_pocket = 10

if money_in_pocket >= cost_of_chocolate:
    print("I have enough money to buy the chocolate!")
```

You need the : (colon) here

This means greater than or equal to

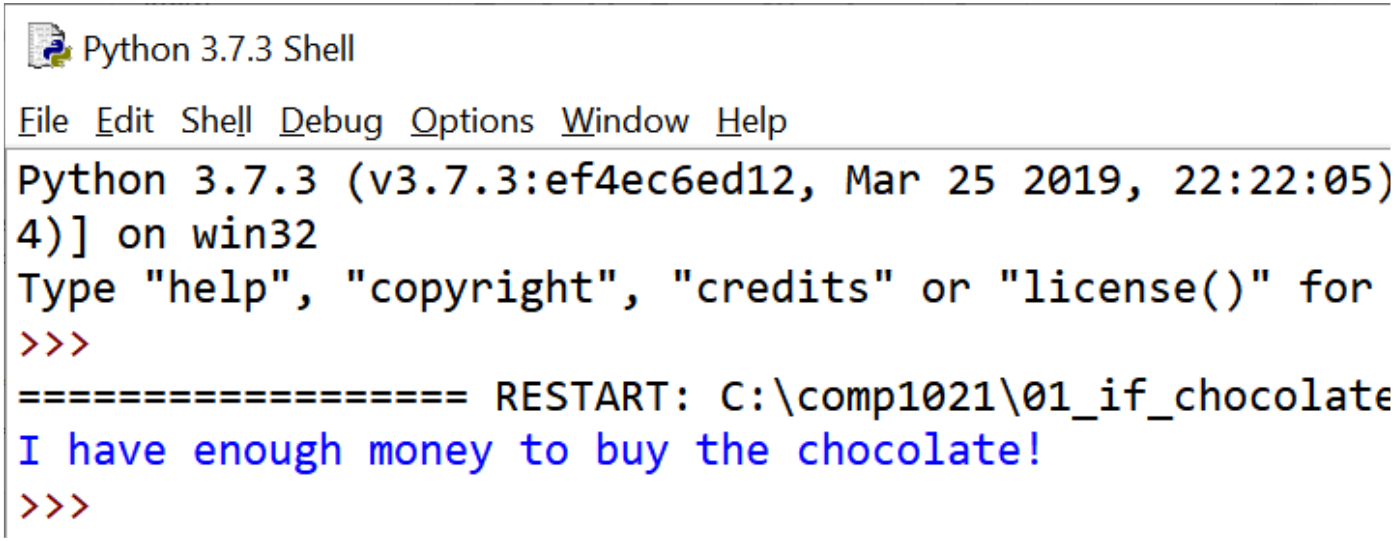
The Flow of the if Statement Example

- We can show the idea using a flowchart, like this:



Running the if Example

- This is what we see when we load and run the program:



- However, what happens if the result of the 'if' is different?

Running the if Example Again

- Let's edit the file and change this line of code:

  ```
  cost_of_chocolate = 10
  ```

  to this:

  ```
  cost_of_chocolate = 12
  ```

- When we load and run the code, it looks like this:

```
============= RESTART: C:\comp1021\02_if
>>>
```

- As you can see, nothing is produced by the program!

The if Condition

- Python decides whether to run the code inside the if statement by evaluating the *condition*
- In our example, the condition is the one shown below (enough money to buy chocolate?):

```
. . .
if money_in_pocket >= cost_of_chocolate:
    . . .
```

- If the condition is true, Python will run the code inside the if statement; otherwise, Python will skip the code

Using Comparison Operators

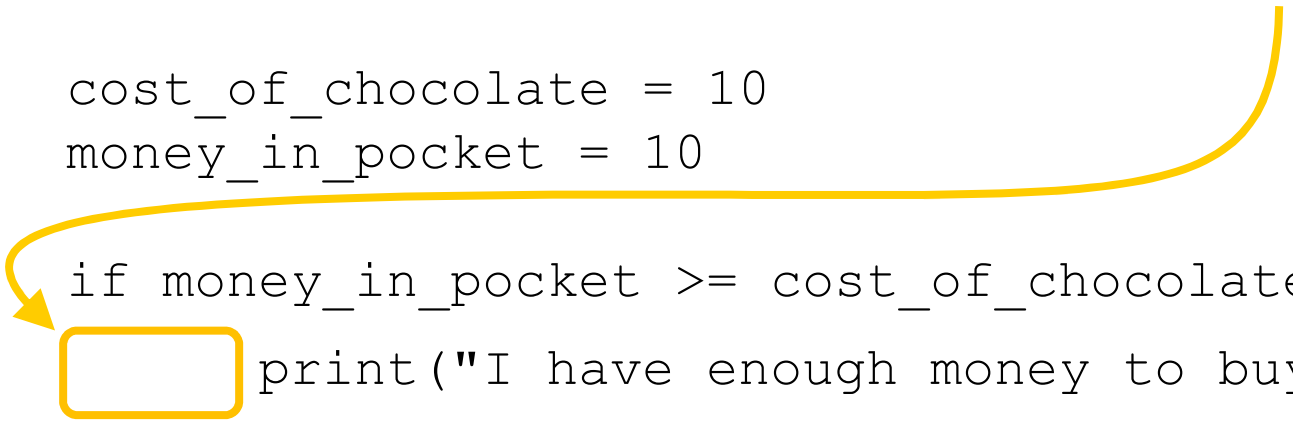
- You can do the following comparisons:

<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
!=	not equal to

- You can also use *and* *or* and *not*, discussed in another presentation

You Must Use Indentation

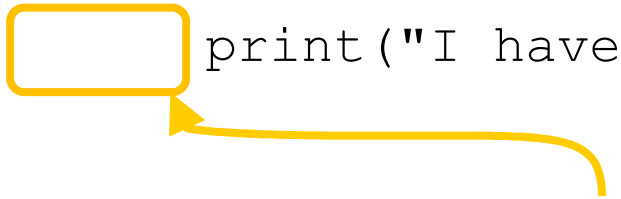
- In programming, ‘indentation’ means ‘moving the lines of code to the right, to the appropriate place’
- For Python code, indentation is **very important**!
- For example, there MUST be indentation here



```
cost_of_chocolate = 10
money_in_pocket = 10

if money_in_pocket >= cost_of_chocolate:
    print("I have enough money to buy the chocolate!")
```

You Must Use Indentation

```
cost_of_chocolate = 10
money_in_pocket = 10

if money_in_pocket >= cost_of_chocolate:
    print("I have enough money to buy the chocolate!")
```



- Without this indentation, the program won't work!
- Although this sounds like trouble, it means that when you look at any Python code, it is easier to understand

How to Do the Indentation?

```
if money_in_pocket >= cost_of_chocolate:
    print("I have enough money to buy the chocolate!")
```

- How far should you move a line of code for each ‘unit’ of indentation?
- Usually, people simply press the Tab key
- When you press the Tab key the IDLE editor adds 4 spaces

Indentation in IDLE

- The IDLE editor which we use is helpful
  – for example, if you type

  ```
  if x > 4:
  ```

  and then press enter, IDLE will understand that you need to have some indentation on the next line and will automatically add it for you!

Extending the if Statement

- In English you might say:

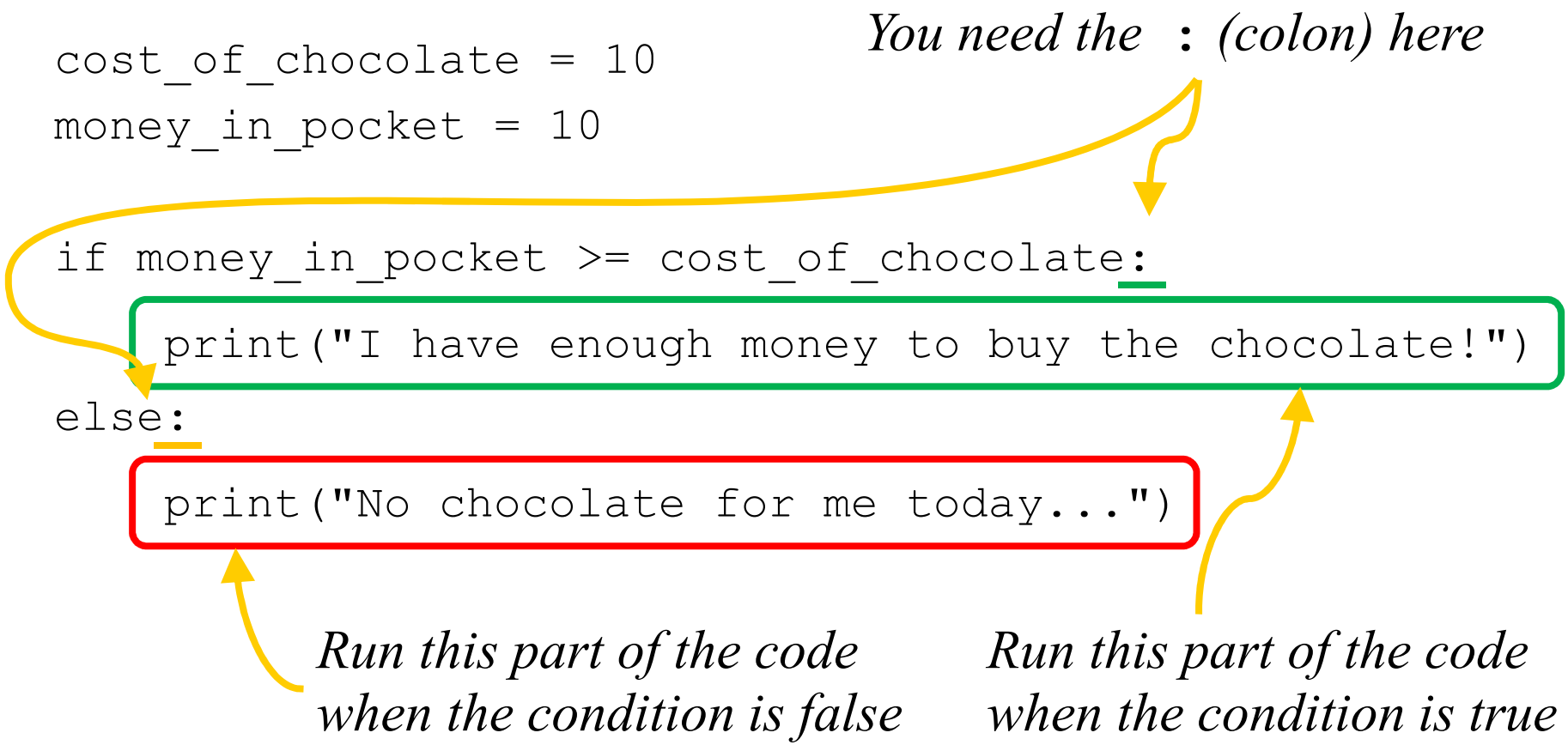
  "**if** *something* **otherwise** *something else*"

- For example:

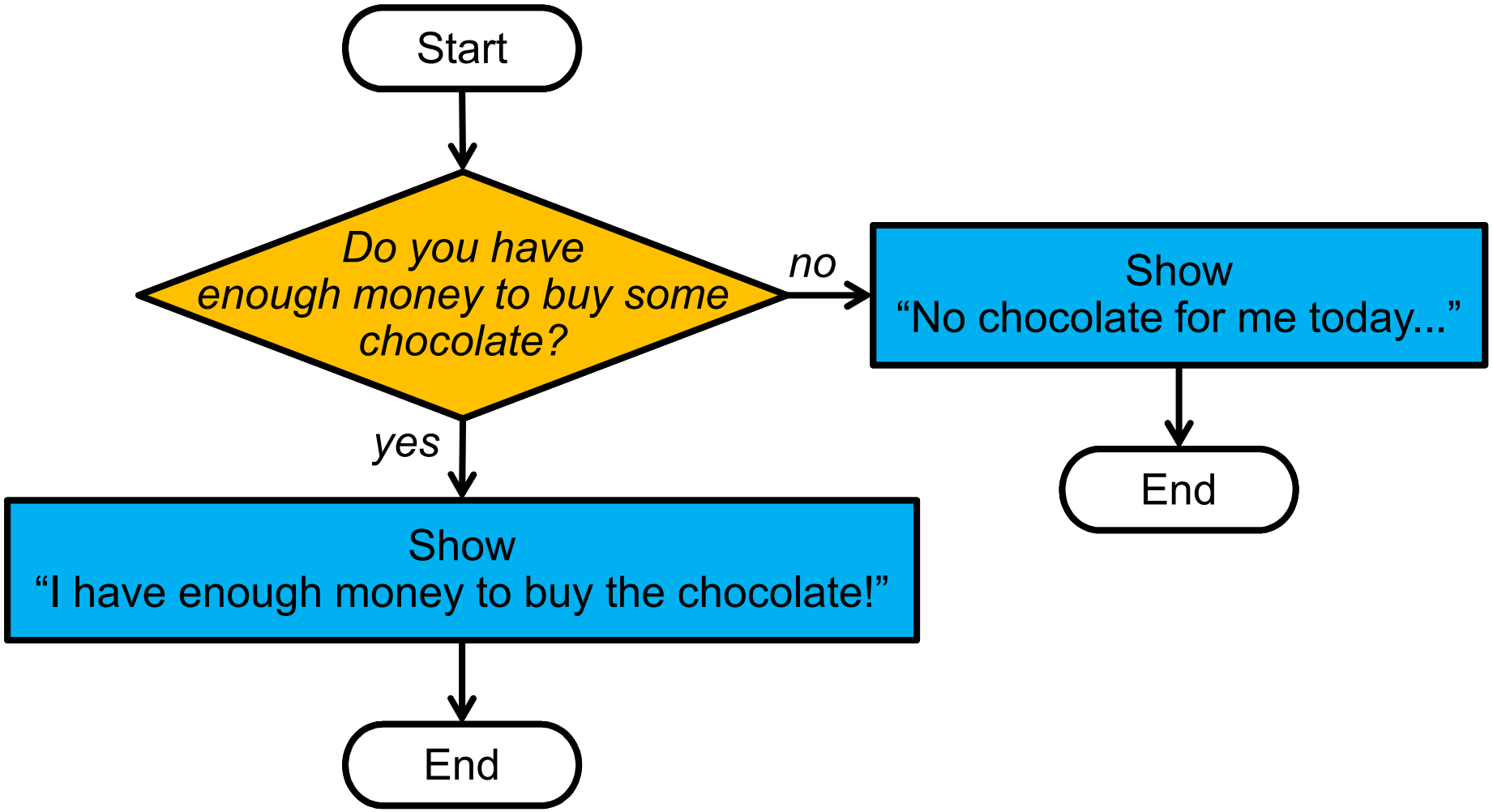
  "**if** I have a million dollars I am rich **otherwise** I am poor"

- We can do the same thing in Python by using the word 'else'
- An example is shown on the next slide

The if … else Example

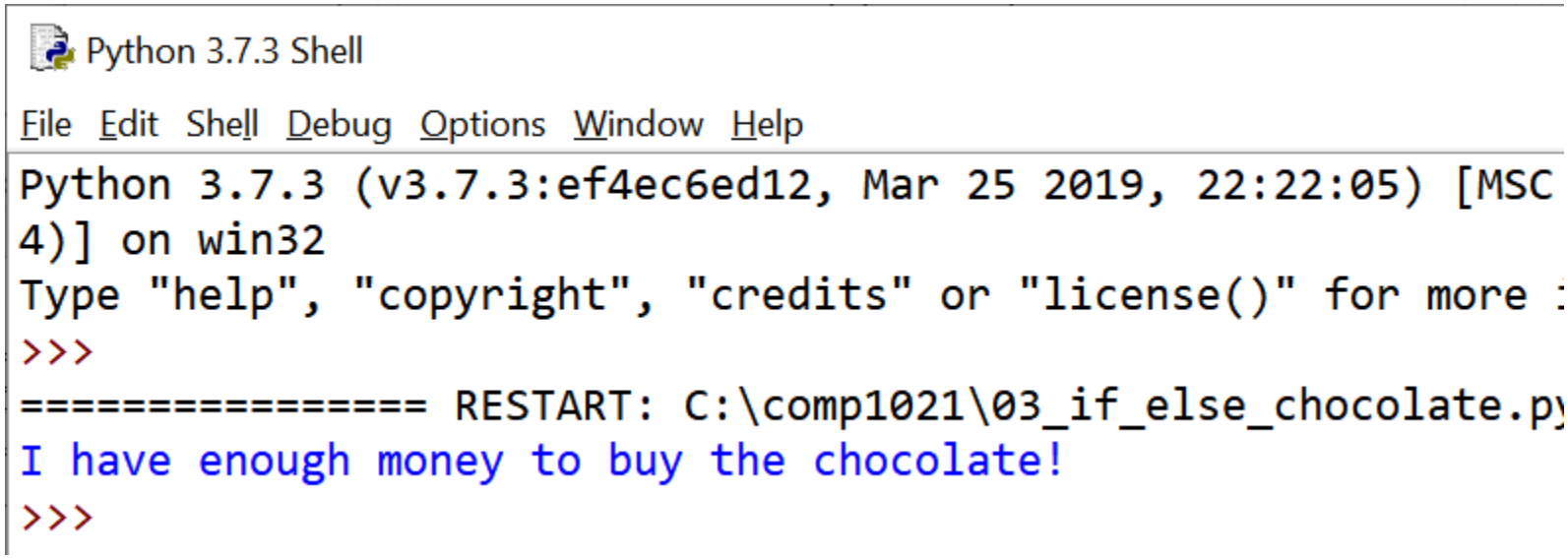


The Flow of the if … else Example



Running the if … else Example

- This is what we see when we run the program:



- However, what happens if the result of the 'if' is different?

Running the if … else Example Again

- Let's edit the file and change this line of code:

  ```
  cost_of_chocolate = 10
  ```

  to this:

  ```
  cost_of_chocolate = 12
  ```

- When we run the code, it looks like this:

```
================ RESTART: C:\comp1021\03_if_else
No chocolate for me today...
>>>
```

- This is much nicer than showing nothing!

Multiple if Statements

- Sometimes it is useful to do a second test if the first test fails
- Here is an example:

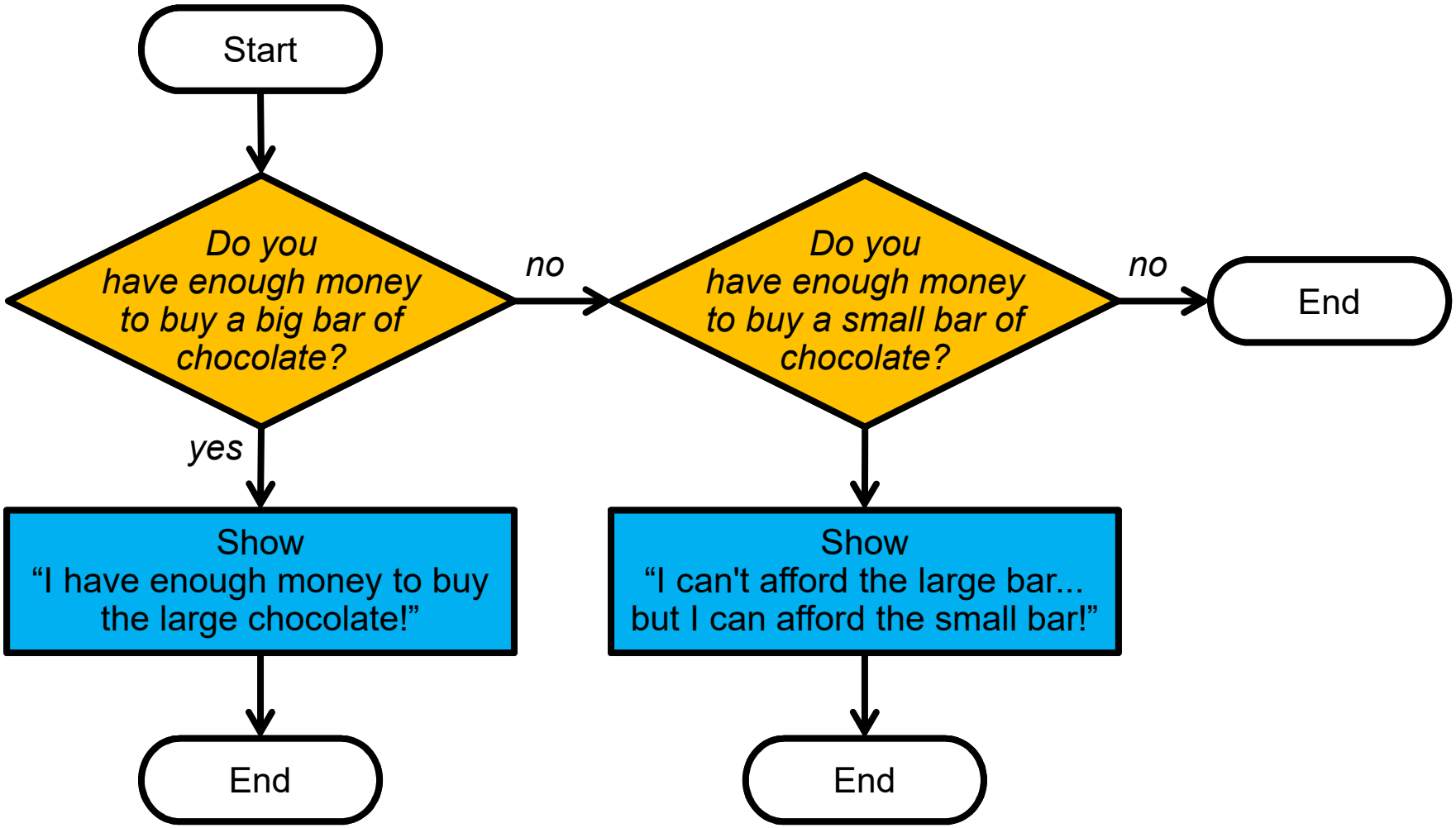
```
cost_of_large_chocolate_bar = 12
cost_of_small_chocolate_bar = 8
money_in_pocket = 10

if money_in_pocket >= cost_of_large_chocolate_bar:
    print("I have enough money to buy
          the large chocolate!")
else:
    if money_in_pocket >= cost_of_small_chocolate_bar:
        print("I can't afford the large bar...
              but I can afford the small bar!")
```

The if … elif Statement

- Instead of using many if statements we can use the elif statement, for example:

```
cost_of_large_chocolate_bar = 12
cost_of_small_chocolate_bar = 8
money_in_pocket = 10

if money_in_pocket >= cost_of_large_chocolate_bar:
    print("I have enough money to buy
          the large chocolate!")
elif money_in_pocket >= cost_of_small_chocolate_bar:
    print("I can't afford the large bar...
          but I can afford the small bar!")
```

The Flow of the if … elif Example

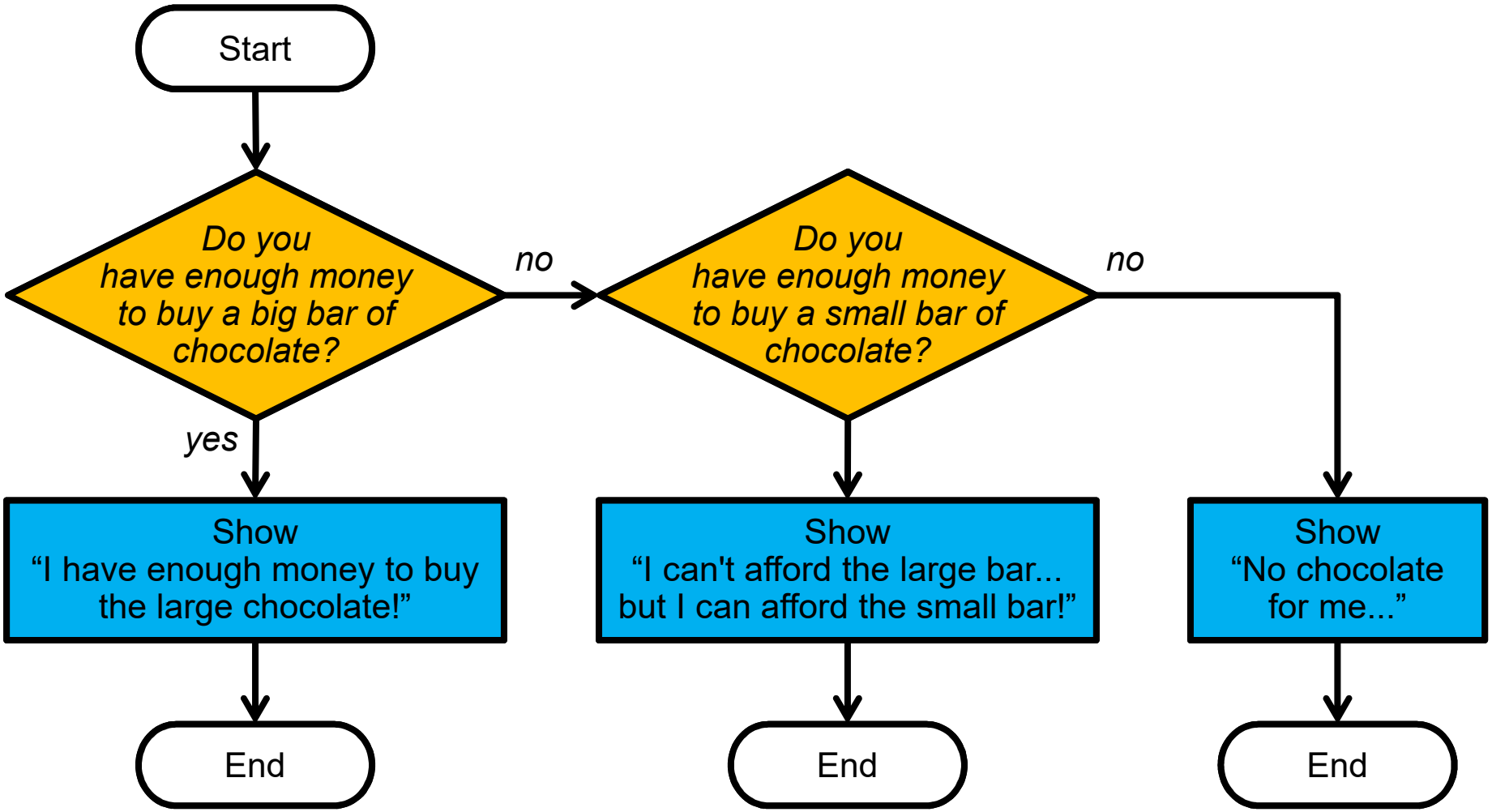


Using if … elif … else

- If both tests fail, our previous example doesn't display anything
- Let's fix that by adding an else:

```
cost_of_large_chocolate_bar = 12
cost_of_small_chocolate_bar = 8
money_in_pocket = 10

if money_in_pocket >= cost_of_large_chocolate_bar:
    print("I have enough money to buy
          the large chocolate!")
elif money_in_pocket >= cost_of_small_chocolate_bar:
    print("I can't afford the large bar...
          but I can afford the small bar!")
else:
    print("No chocolate for me...")
```

The Flow of the if…elif…else Example



An Example with Many elif 1/2

- You can have as many elif parts as you like
- Here's an example which 'converts' a number into English:

```
value = input("Enter a number from 0 to 9: ")
value = int(value)
if value == 0:
    number_in_english = "Zero"
elif value == 1:
    number_in_english = "One"
elif value == 2:
    number_in_english = "Two"
elif value == 3:
    number_in_english = "Three"
```

This means 'equal to'

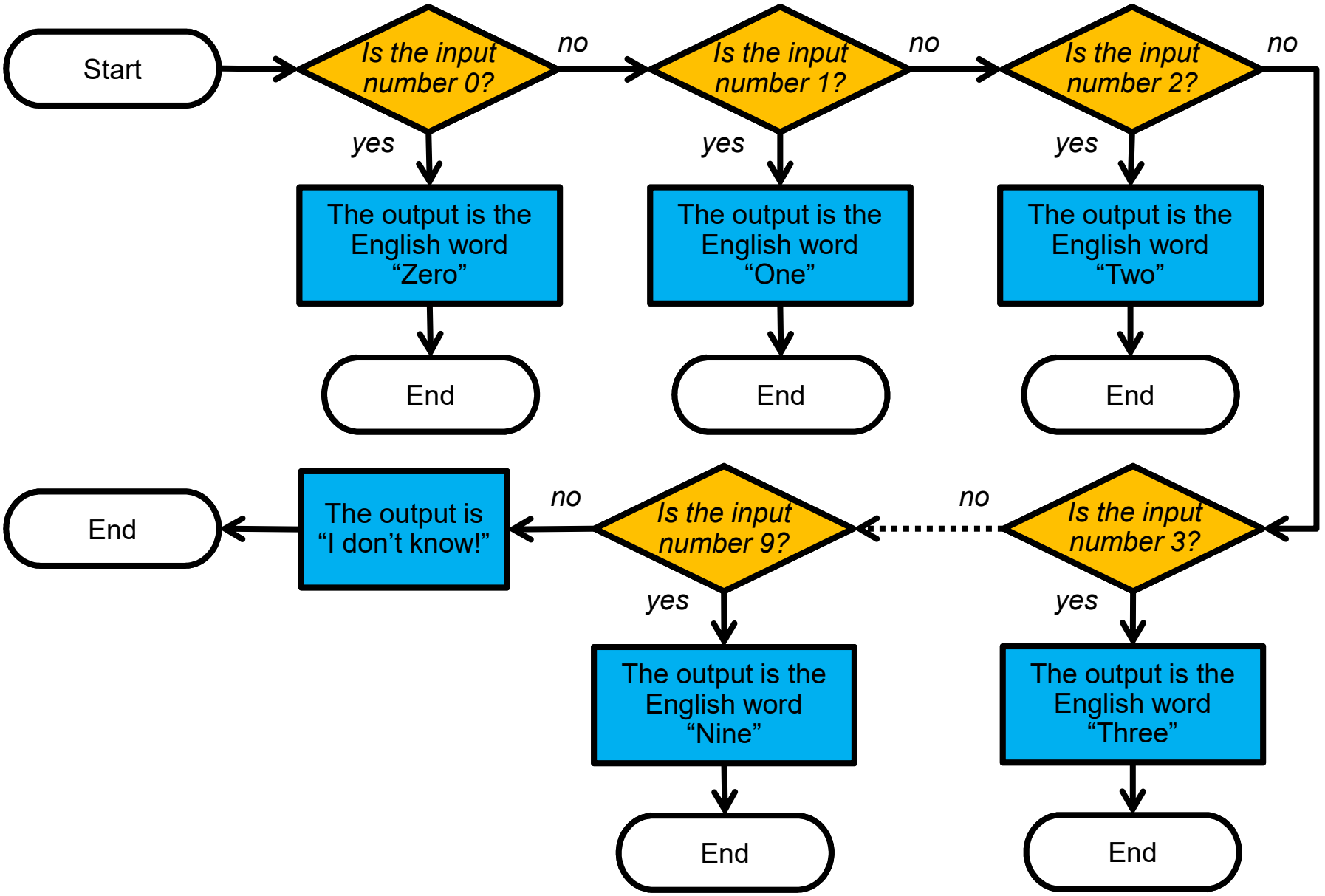
A space has been added at the end of the question so that a space is shown after the text `Enter a number from 0 to 9:` *– the space makes the resulting text look nicer*

An Example with Many elif 2/2

```
elif value == 4:
    number_in_english = "Four"
elif value == 5:
    number_in_english = "Five"
elif value == 6:
    number_in_english = "Six"
elif value == 7:
    number_in_english = "Seven"
elif value == 8:
    number_in_english = "Eight"
elif value == 9:
    number_in_english = "Nine"
else:
    number_in_english = "I don't know!"
print("The number in English is", number_in_english)
```

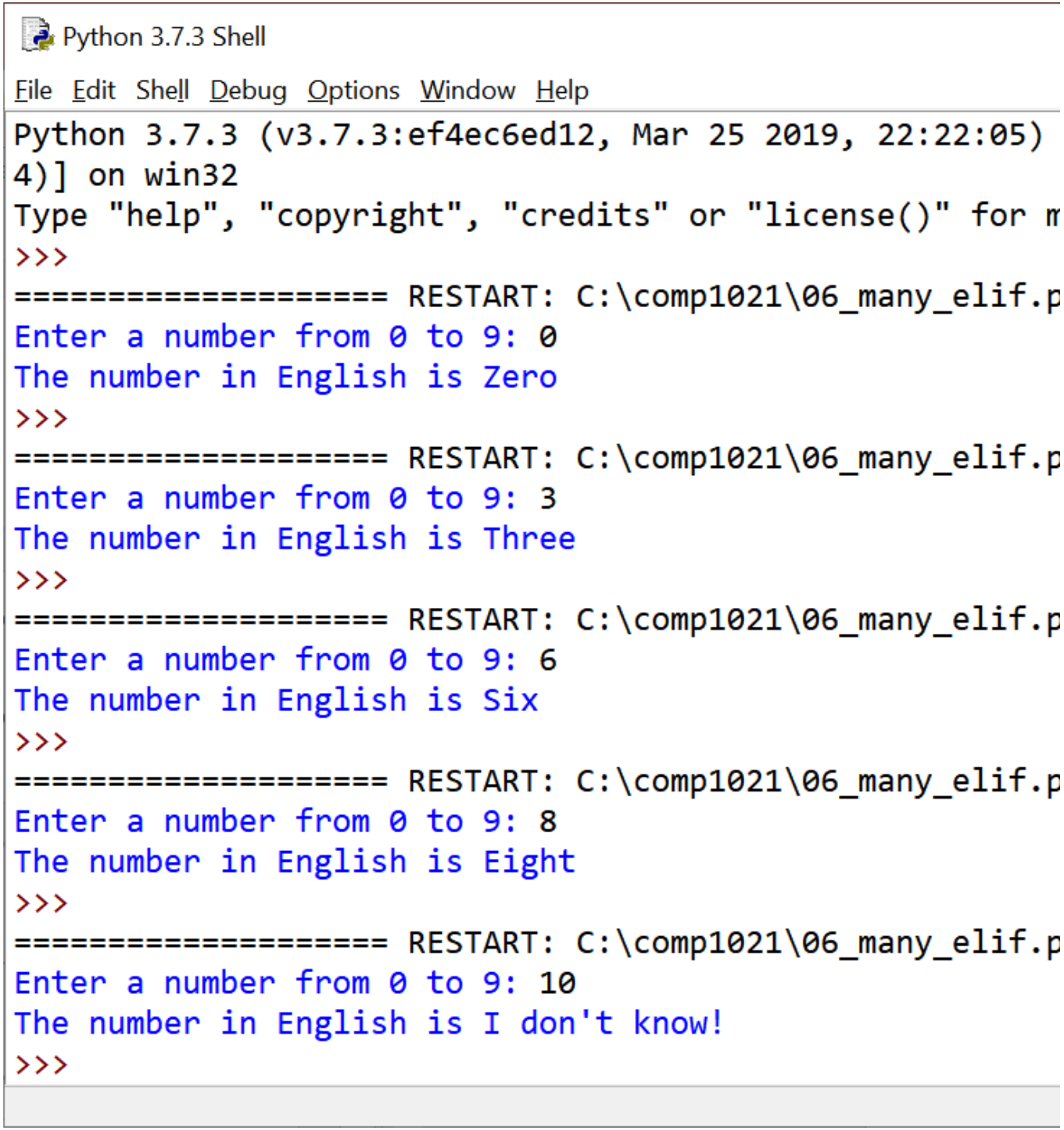
- `print()` always adds a space after each thing it prints
- So in this case a space will be automatically added after the text `The number in English is`

The Flow of Using Many elif



Running the Program

- Here's some examples of running the program



The Basic Pattern of if

```
if ...condition...  :
    ...some code...
elif ...condition...  :
    ...some code...
else :
    ...some code...
```

- The *if* gets things started
- There's >=0 *elif*
- You can have as many *elif* as you want
- There's zero or 1 *else*
- If *else* is used then it is always at the end

Nested if Statements

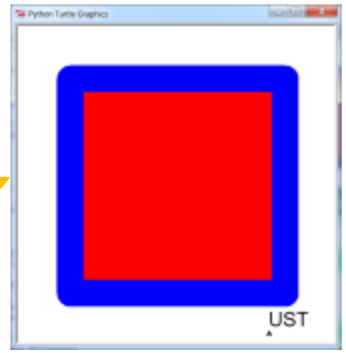
- If you want to you can put an if statement inside another if statement
- When you do that it is called a *nested if*
- The next slide shows the 'flow structure' of an example using turtle graphics and nested if statements

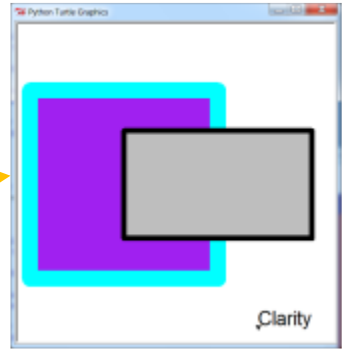
```
if ...condition... :
    ...some code...
    if ...condition... :
        ...some code...
    ...some code...
```

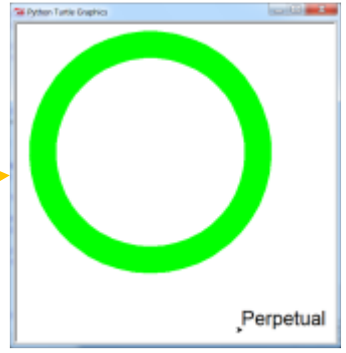
```
Beautiful Modern Art!

This program will display some beautiful
modern art, according to your choices.

Please choose one of the following.

1 - using squares/rectangles
2 - using turtle.circle

Which one do you want?
```

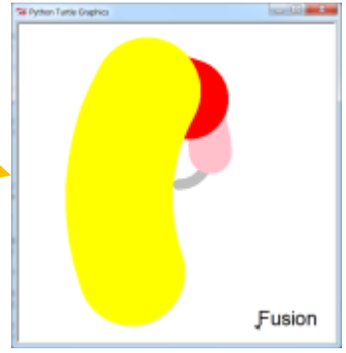
```
Please choose one of the following.

a - simple art using squares/rectangles
b - advanced art using squares/rectangles

Which one do you want?
```

```
Please choose one of the following.

a - simple art using turtle.circle
b - advanced art using turtle.circle

Which one do you want?
```





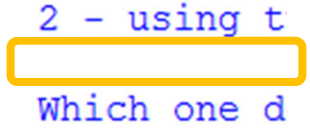




The Code 1/4

```
import turtle

print("Beautiful Modern Art!")
print()
print("This program will display some beautiful")
print("modern art, according to your choices.")
print()
print("Please choose one of the following.")
print()
print("1 - using squares/rectangles")
print("2 - using turtle.circle")
print()

choice = input("Which one do you want? ")
```

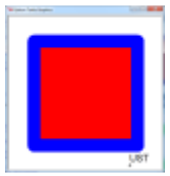
This code prints an empty line, i.e.



The Code 2/4

Single indentation (4 spaces) for these areas

```
if choice == "1":
    print("Please choose one of the following.")
    print()
    print("a - simple art using squares/rectangles")
    print("b - advanced art using squares/rectangles")
    print()
    second_choice = input("Which one do you want? ")

    if second_choice == "a":
        ... draw a square ...
    elif second_choice == "b":
        ... draw a square and rectangle ...
    else:
        print("You need to enter a or b!")
```



Double indentation (i.e. 8 spaces) for these areas

The Code 3/4

```
elif choice == "2":
    print("Please choose one of the following.")
    print()
    print("a - simple art using turtle.circle")
    print("b - advanced art using turtle.circle")
    print()
    second_choice = input("Which one do you want? ")

    if second_choice == "a":
        ... draw a circle ...
    elif second_choice == "b":
        ... draw sections of circles ...
    else:
        print("You need to enter a or b!")
```

The Code 4/4

```
else:
    print("You need to enter 1 or 2!")
```

- If the user doesn't enter a '1' or a '2' when replying to the first question, this part will be executed