

CONDITION EXECUTION

Agenda

- How to use “if condition” in conditional Structures
- How to use “elif” condition
- Nested IF Statements
- Debugging

Condition Execution

- Boolean Expression: A Boolean expression is an expression that is either true or false. The following examples use the operator `==`, which compares two operands and produces `True` if they are equal and `False` otherwise:

A screenshot of a Python 3.6.1 Shell window. The title bar is yellow and says "Python 3.6.1 Shell". Below it is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area is a dark blue terminal with white text. It shows the Python version and build information: "Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32". It then prompts the user to type "copyright", "credits", or "license()". The user has entered two expressions: ">>> 5 == 5" which returns "True", and ">>> 5 == 6" which returns "False". The prompt ">>>" is followed by a cursor. The status bar at the bottom right shows "Ln: 7 Col: 4".

```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 5 == 5
True
>>> 5 == 6
False
>>> |
```

Ln: 7 Col: 4

Continue...

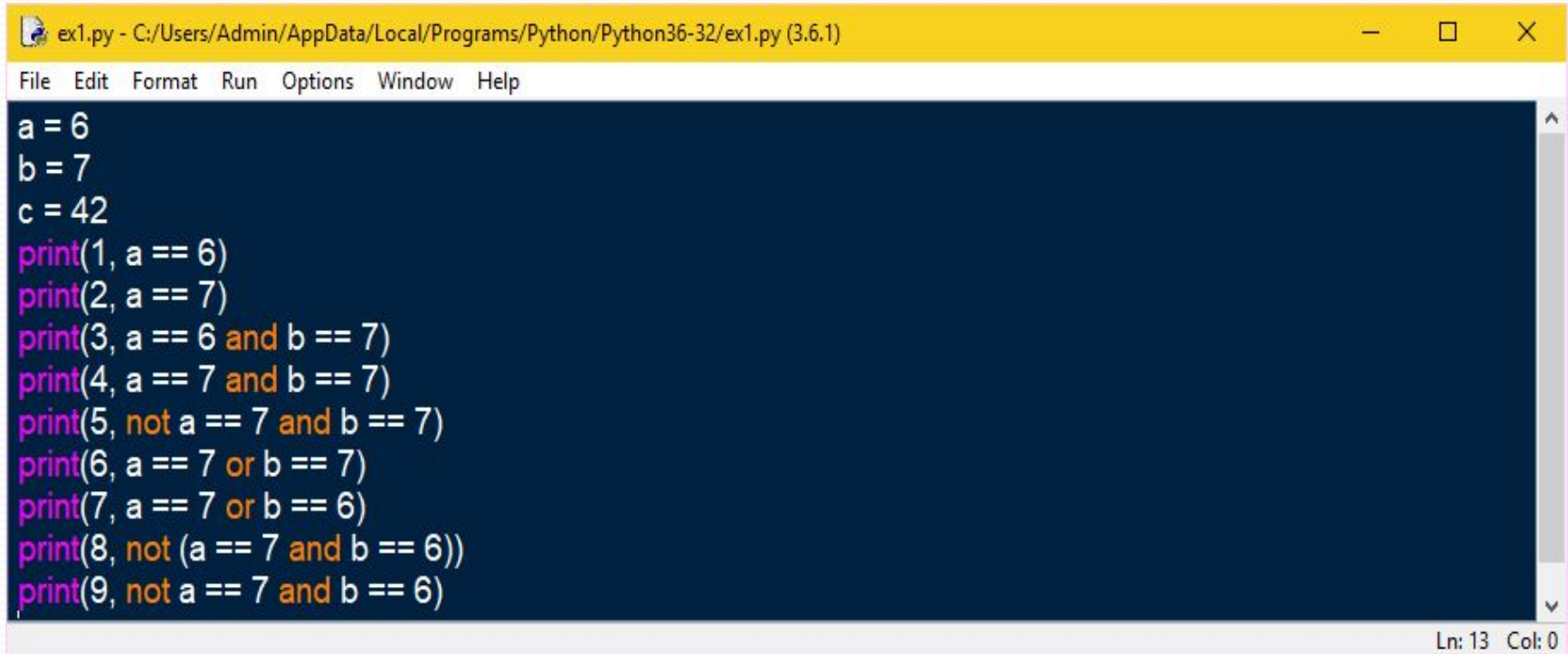
- True and False are special values that belong to the type bool; they are not Strings

A screenshot of a Python 3.6.1 Shell window. The title bar is yellow and says "Python 3.6.1 Shell". Below it is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area is a dark blue terminal with white text. It shows the Python version and build information: "Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32". It then prompts the user to type "copyright", "credits", or "license()". The user has entered two commands: ">>> type(True)" and ">>> type(False)". Both commands return the output "<class 'bool'>". The status bar at the bottom right shows "Ln: 7 Col: 4".

```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> type(True)
<class 'bool'>
>>> type(False)
<class 'bool'>
Ln: 7 Col: 4
```

Continue...

- Some Boolean expressions are given below
- Example 1:

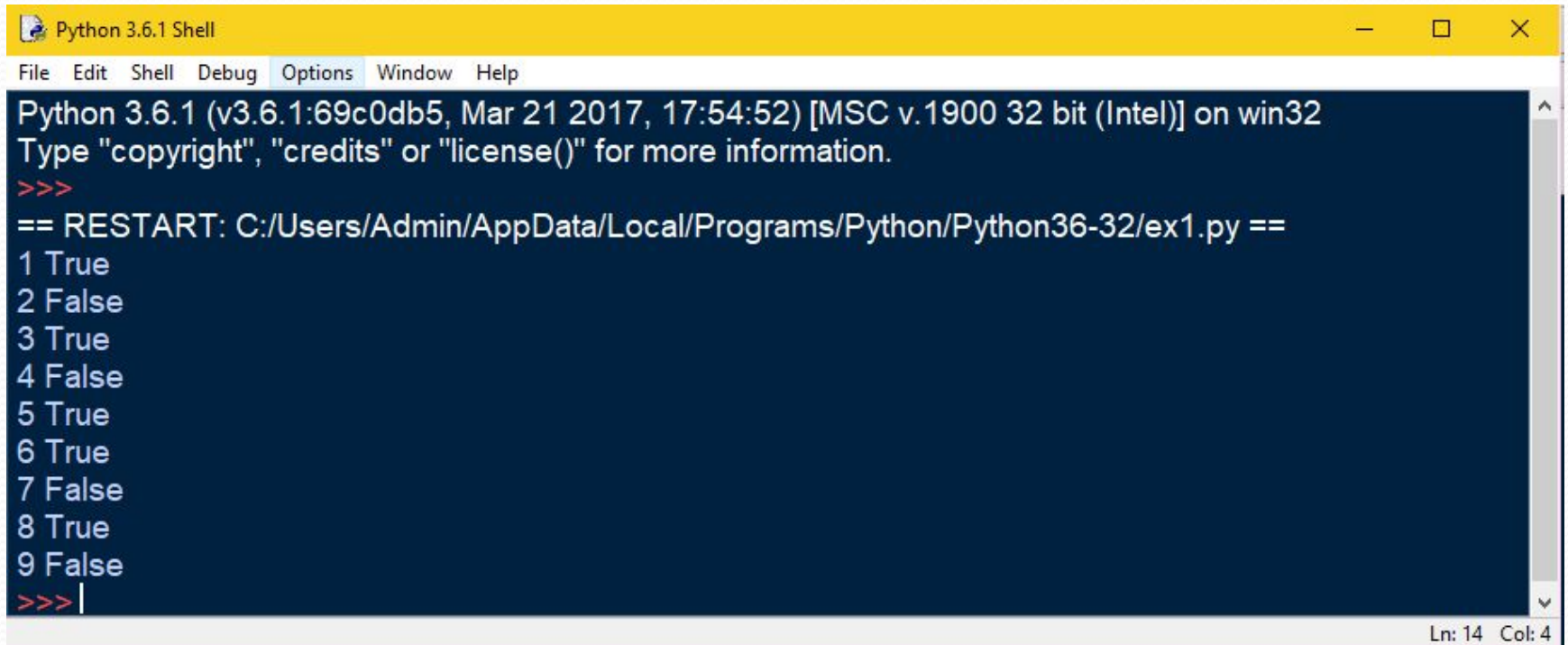


The screenshot shows a Python IDE window titled "ex1.py - C:/Users/Admin/AppData/Local/Programs/Python/Python36-32/ex1.py (3.6.1)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main editor area has a dark blue background and contains the following Python code:

```
a = 6
b = 7
c = 42
print(1, a == 6)
print(2, a == 7)
print(3, a == 6 and b == 7)
print(4, a == 7 and b == 7)
print(5, not a == 7 and b == 7)
print(6, a == 7 or b == 7)
print(7, a == 7 or b == 6)
print(8, not (a == 7 and b == 6))
print(9, not a == 7 and b == 6)
```

The status bar at the bottom right indicates "Ln: 13 Col: 0".

Continue...



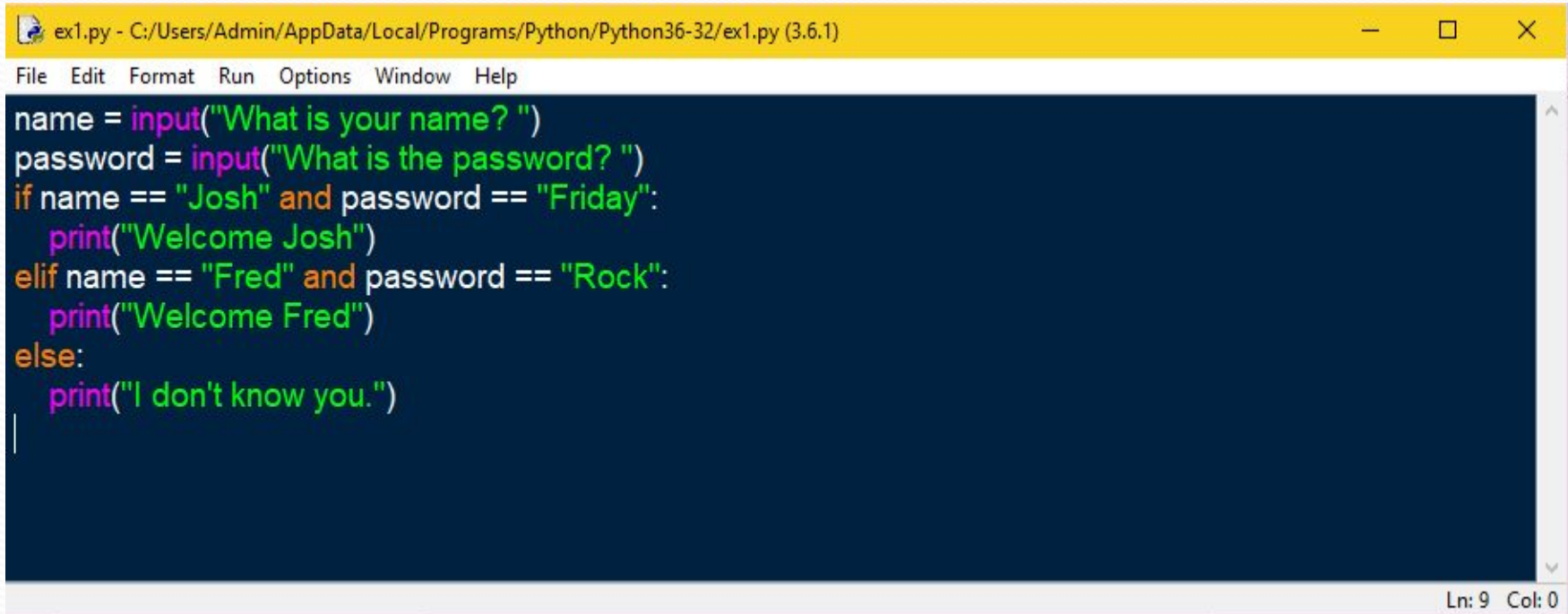
The image shows a screenshot of a Python 3.6.1 Shell window. The window has a yellow title bar with the text "Python 3.6.1 Shell" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area is a dark blue console with white text. It displays the Python version and build information: "Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32". It then prompts the user to type "copyright", "credits", or "license()". After the user presses Enter, it shows "==" followed by the file path "C:/Users/Admin/AppData/Local/Programs/Python/Python36-32/ex1.py ==". The script then executes a series of nine lines, each starting with a line number followed by a boolean value: "1 True", "2 False", "3 True", "4 False", "5 True", "6 True", "7 False", "8 True", and "9 False". The prompt ">>>" is visible at the bottom of the console. A status bar at the bottom right shows "Ln: 14 Col: 4".

```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python36-32/ex1.py ==
1 True
2 False
3 True
4 False
5 True
6 True
7 False
8 True
9 False
>>> |
```

Ln: 14 Col: 4

Continue....

- Example 2: This program asks a user for a name and a password it then check them to make sure that the user is allowed in.
- Program name: Sample.py

A screenshot of a Python IDE window titled "ex1.py - C:/Users/Admin/AppData/Local/Programs/Python/Python36-32/ex1.py (3.6.1)". The window has a yellow title bar and a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main area is a dark blue editor with Python code. The code prompts for a name and password, then checks if the name is "Josh" and password is "Friday" (welcome) or "Fred" and password is "Rock" (welcome). Otherwise, it says "I don't know you." The status bar at the bottom right shows "Ln: 9 Col: 0".

```
ex1.py - C:/Users/Admin/AppData/Local/Programs/Python/Python36-32/ex1.py (3.6.1)
File Edit Format Run Options Window Help
name = input("What is your name? ")
password = input("What is the password? ")
if name == "Josh" and password == "Friday":
    print("Welcome Josh")
elif name == "Fred" and password == "Rock":
    print("Welcome Fred")
else:
    print("I don't know you.")
|
```

Ln: 9 Col: 0

Logical operators

- There are three logical operators: and, or, and not. The semantics (meaning) of these operators is similar to their meaning in English.
- For example $x > 0$ and $x < 10$, is true only if x is greater than 0 and less than 10.
- $n \% 2 == 0$ or $n \% 3 == 0$, is true if either of the conditions is true, that is, if the number is divisible by 2 or 3.
- Finally, the not operator negates a Boolean expression, so $\text{not } (x > y)$ is true if $x > y$ is false, that is, if x is less than or equal to y .

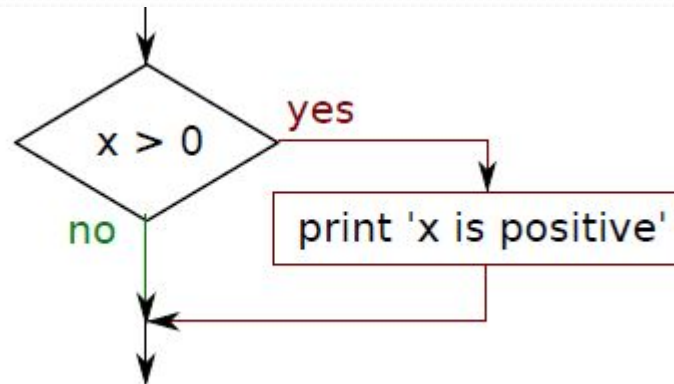
Conditional execution

- In order to write useful programs, we almost always need the ability to check conditions and change the behavior of the program accordingly. Conditional statements give us this ability.
- The simplest form is the if statement
- An **if statement** consists of a boolean expression followed by one or more statements.

If Statement

- An **if statement** consists of a boolean expression followed by one or more statements.
- The Boolean expression after the if statement is called the condition. We end the if statement with a colon character (:) and the line(s) after the if statement are indented.

```
if x > 0 :  
    print ("x is positive")
```

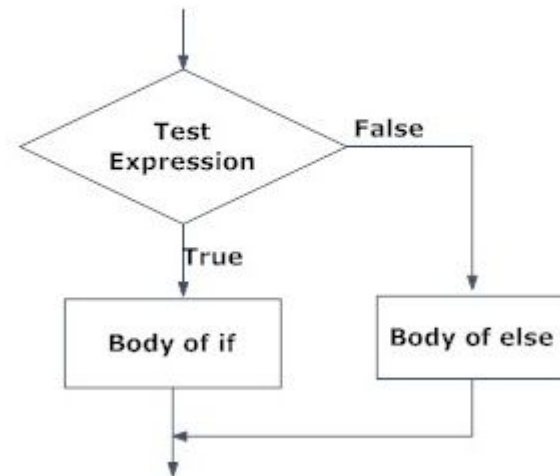


If else Statement

- The if..else statement evaluates test expression and will execute body of if only when test condition is True.
- If the condition is False, body of else is executed. Indentation is used to separate the blocks.

```
if test expression:  
    Body of if  
else:  
    Body of else
```

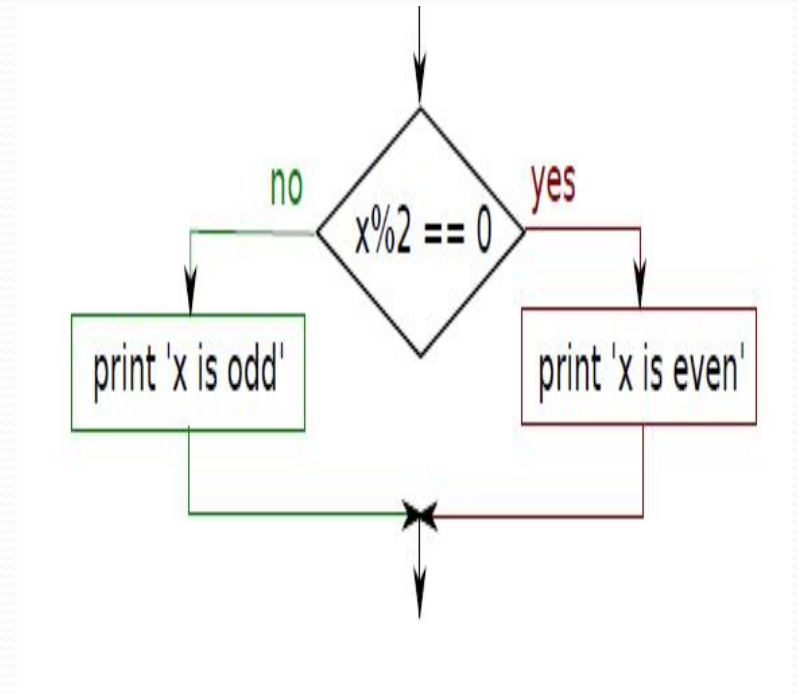
Python if..else Flowchart



Continue...

```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
Python 3.6.3 (v3.6.3:2c5fed8, Oct 3 2016) on win32
Type "copyright", "credits" or "license()" for more
>>> x=3
>>> if x%2==0:
    print("Even")
else:
    print("Odd")

Odd
>>>
```

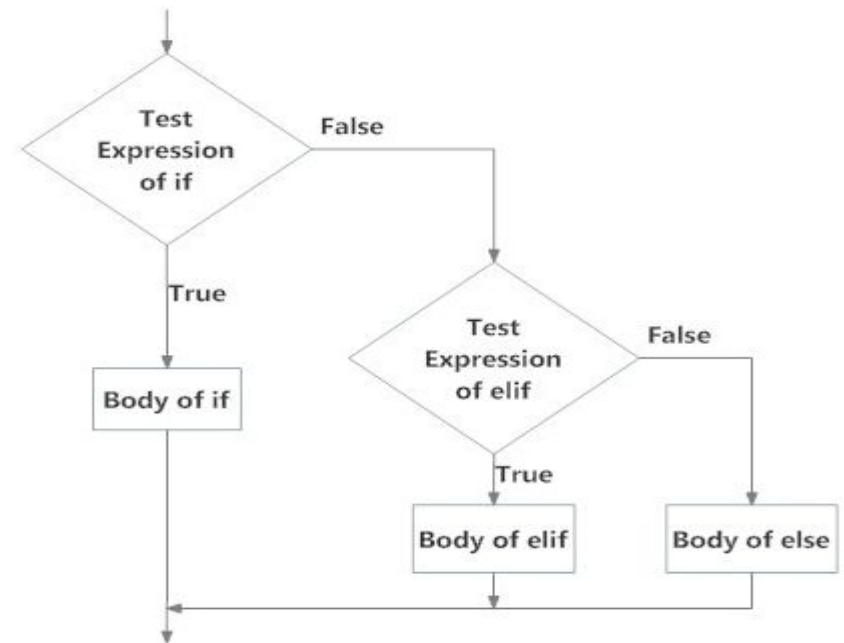


If..elif..else

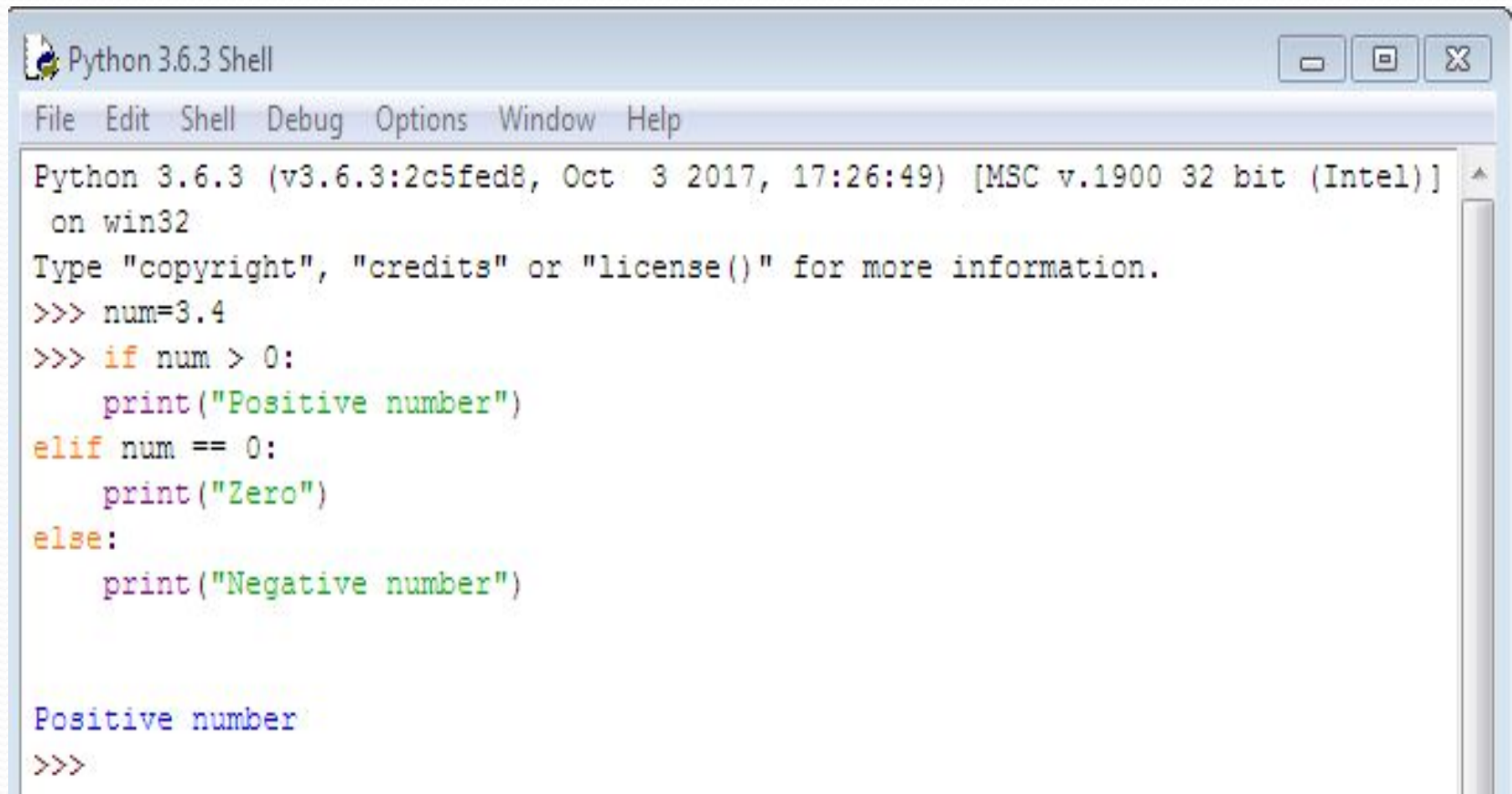
- The elif is short for else if. It allows us to check for multiple expressions.
- If the condition for if is False, it checks the condition of the next elif block and so on. If all the conditions are False, body of else is executed.

```
if test expression:  
    Body of if  
elif test expression:  
    Body of elif  
else:  
    Body of else
```

Flowchart of if...elif...else



Example

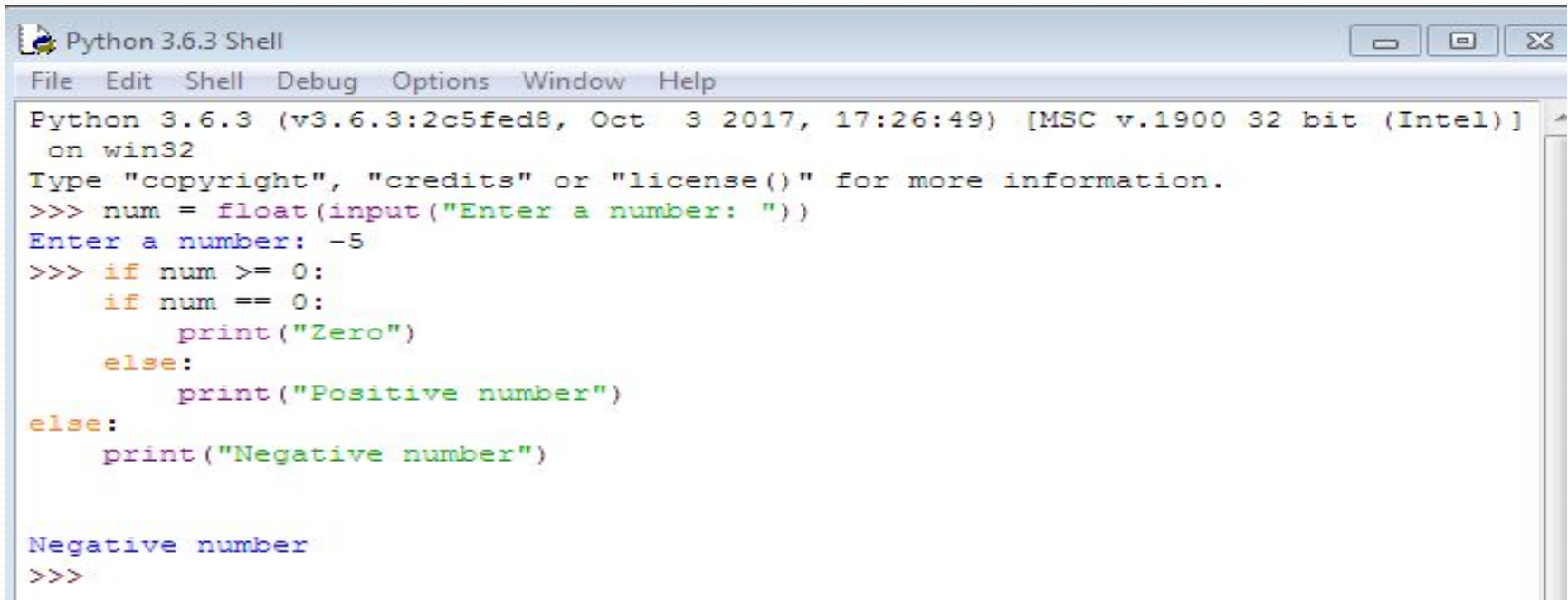


```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
Python 3.6.3 (v3.6.3:2c5fed8, Oct 3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> num=3.4
>>> if num > 0:
    print("Positive number")
elif num == 0:
    print("Zero")
else:
    print("Negative number")

Positive number
>>>
```

Nested If Statement

- We can have a if...elif...else statement inside another if...elif...else statement. This is called nesting in computer programming.

A screenshot of a Python 3.6.3 Shell window. The window has a title bar that says "Python 3.6.3 Shell" and standard Windows window controls (minimize, maximize, close). Below the title bar is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area of the window contains a Python prompt and code. The code defines a variable 'num' as a float from user input. It then uses a nested if statement: an outer 'if' checks if 'num' is greater than or equal to 0. If true, an inner 'if' checks if 'num' is equal to 0. If the inner 'if' is true, it prints "Zero". If the inner 'if' is false (the 'else' branch), it prints "Positive number". If the outer 'if' is false (the 'else' branch), it prints "Negative number". The user has entered "-5", and the output "Negative number" is displayed. The prompt is ready for the next input.

```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> num = float(input("Enter a number: "))
Enter a number: -5
>>> if num >= 0:
    if num == 0:
        print("Zero")
    else:
        print("Positive number")
else:
    print("Negative number")

Negative number
>>>
```


Debugging

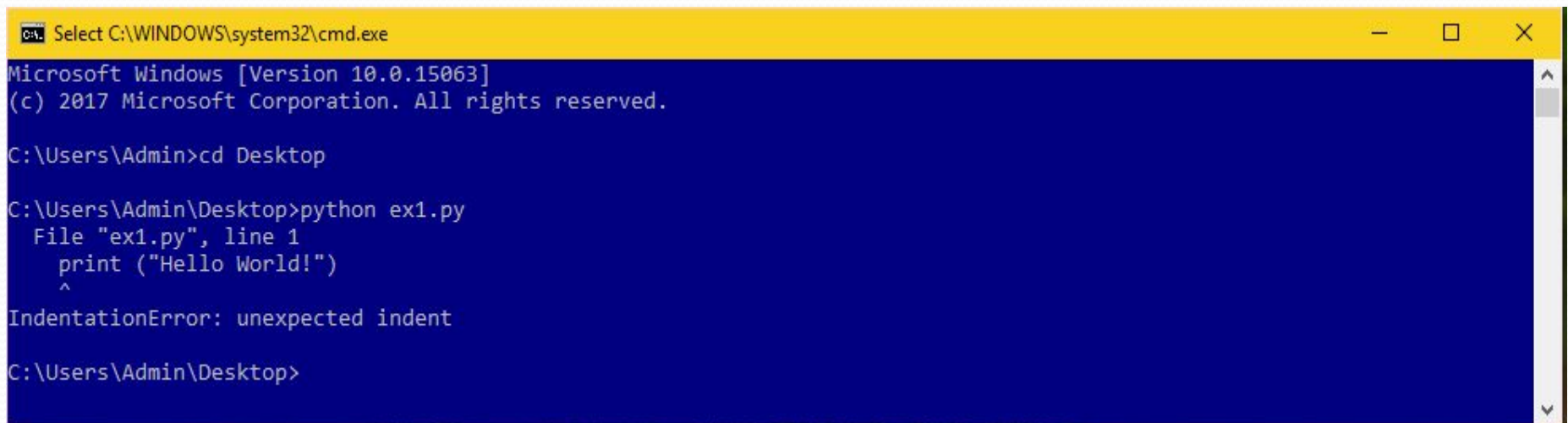
- **Debugging** is the process of finding and resolving defects or problems within the program that prevent correct operation of computer software or a system



A screenshot of a Python IDE window titled "ex1.py - C:\Users\Admin\Desktop\ex1.py (3.6.1)". The window has a yellow title bar and a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main area is a dark blue editor with green text. It contains the following code:

```
print ("Hello World!")
print ("Hello Again")
print ("I like typing this.")
print ("This is fun.")
print ('Yay! Printing.')
print ("I'd much rather you 'not'.")
print ('I "said" do not touch this.')
```

The status bar at the bottom right shows "Ln: 8 Col: 0".



A screenshot of a Windows Command Prompt window titled "Select C:\WINDOWS\system32\cmd.exe". The window has a yellow title bar. The main area is a dark blue console with white text. It shows the following commands and output:

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd Desktop

C:\Users\Admin\Desktop>python ex1.py
File "ex1.py", line 1
    print ("Hello World!")
    ^
IndentationError: unexpected indent

C:\Users\Admin\Desktop>
```