CS65: Introduction to Computer Science

Boolean Expressions
Selection Statements

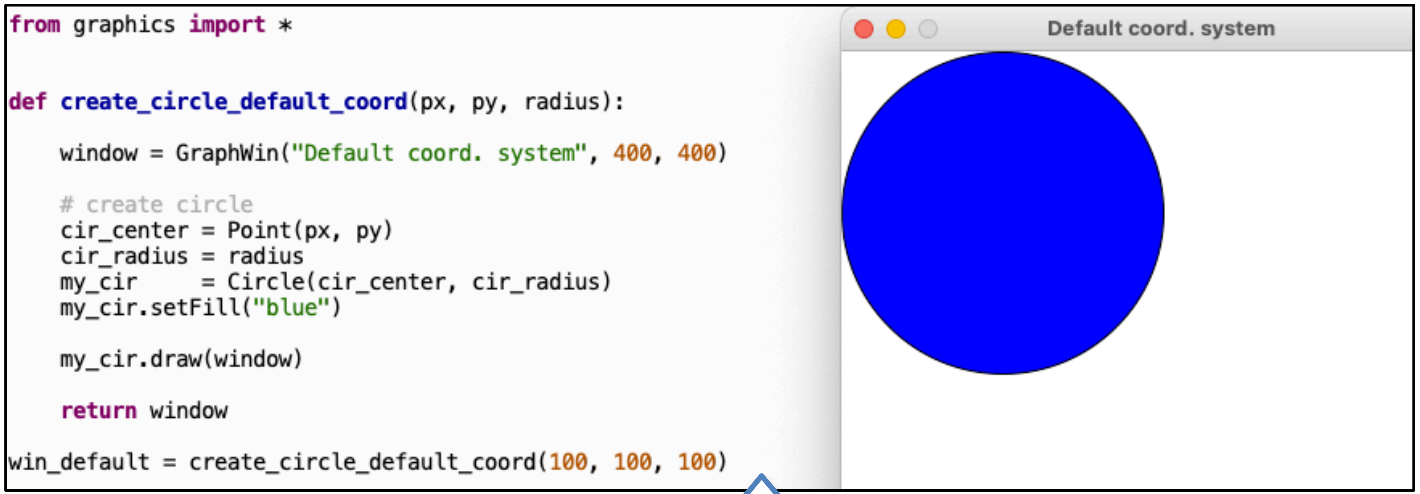
Drake
UNIVERSITY

Md Alimoor Reza
Assistant Professor of Computer Science

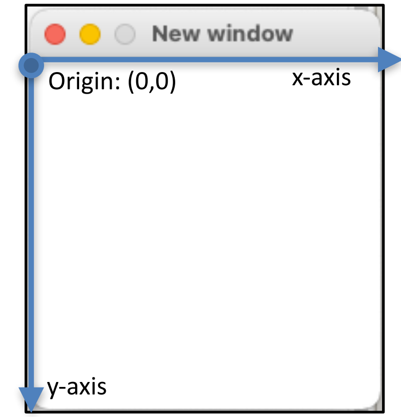
Recap

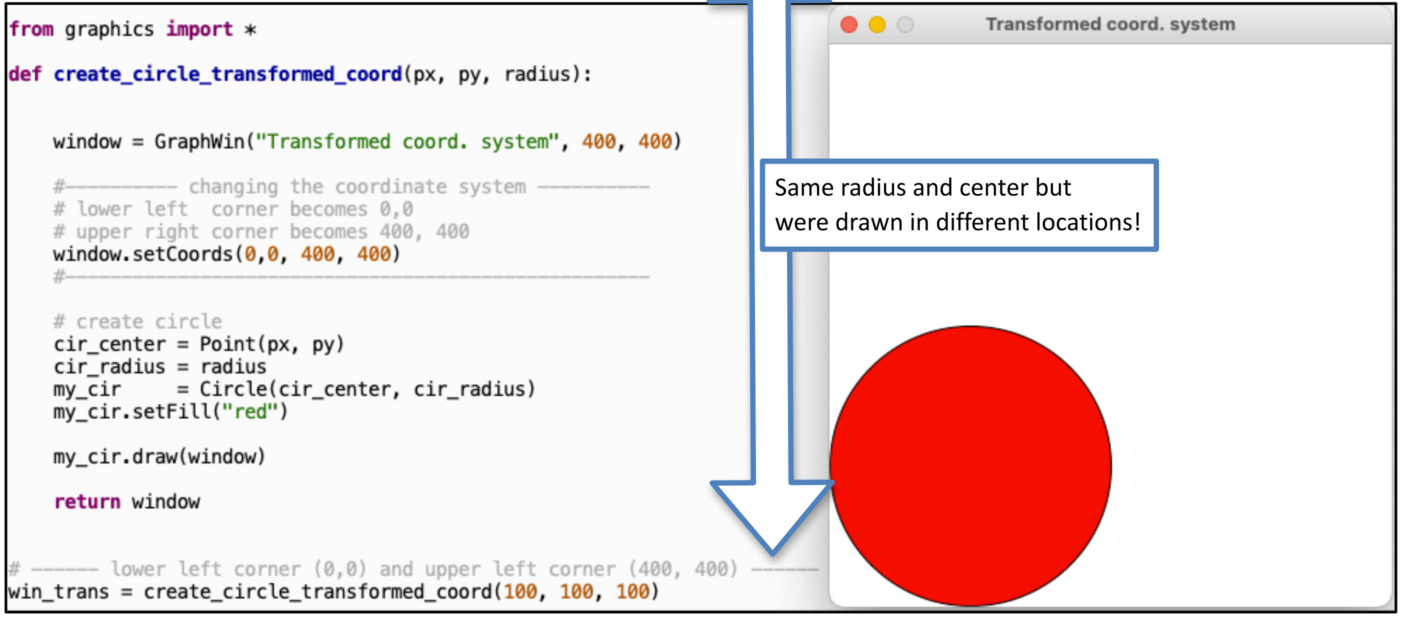
- Drawing shapes inside the window
    - Circle
    - Rectangle
    - Line, Text, and combinations of these shapes
- Changing coordinate system
- Mouse interaction inside graphics window

Recap: Changing coordinate system



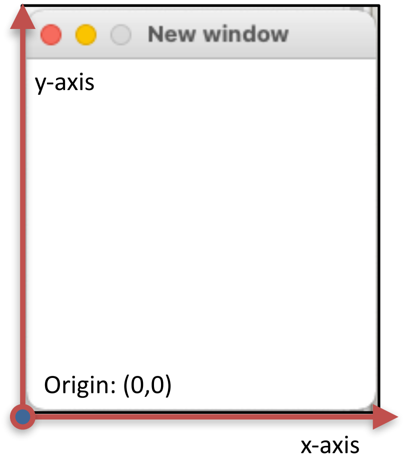
```python
from graphics import *

def create_circle_default_coord(px, py, radius):

    window = GraphWin("Default coord. system", 400, 400)

    # create circle
    cir_center = Point(px, py)
    cir_radius = radius
    my_cir     = Circle(cir_center, cir_radius)
    my_cir.setFill("blue")

    my_cir.draw(window)

    return window

win_default = create_circle_default_coord(100, 100, 100)
```



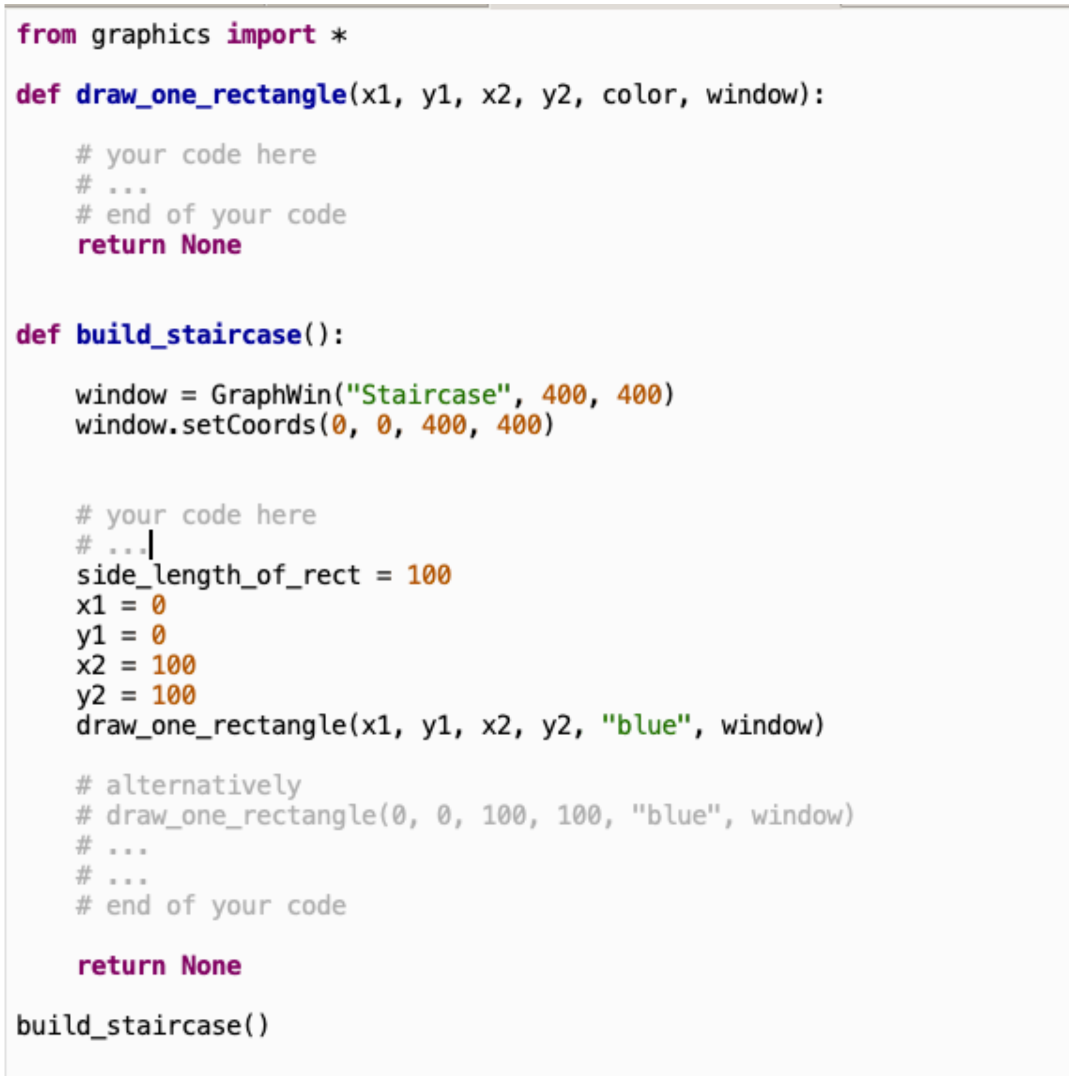


```python
from graphics import *

def create_circle_transformed_coord(px, py, radius):

    window = GraphWin("Transformed coord. system", 400, 400)

    #---------- changing the coordinate system ----------
    # lower left  corner becomes 0,0
    # upper right corner becomes 400, 400
    window.setCoords(0,0, 400, 400)
    #--------------------------------------------------

    # create circle
    cir_center = Point(px, py)
    cir_radius = radius
    my_cir     = Circle(cir_center, cir_radius)
    my_cir.setFill("red")

    my_cir.draw(window)

    return window

# ------ lower left corner (0,0) and upper left corner (400, 400) ------
win_trans = create_circle_transformed_coord(100, 100, 100)
```

Same radius and center but were drawn in different locations!



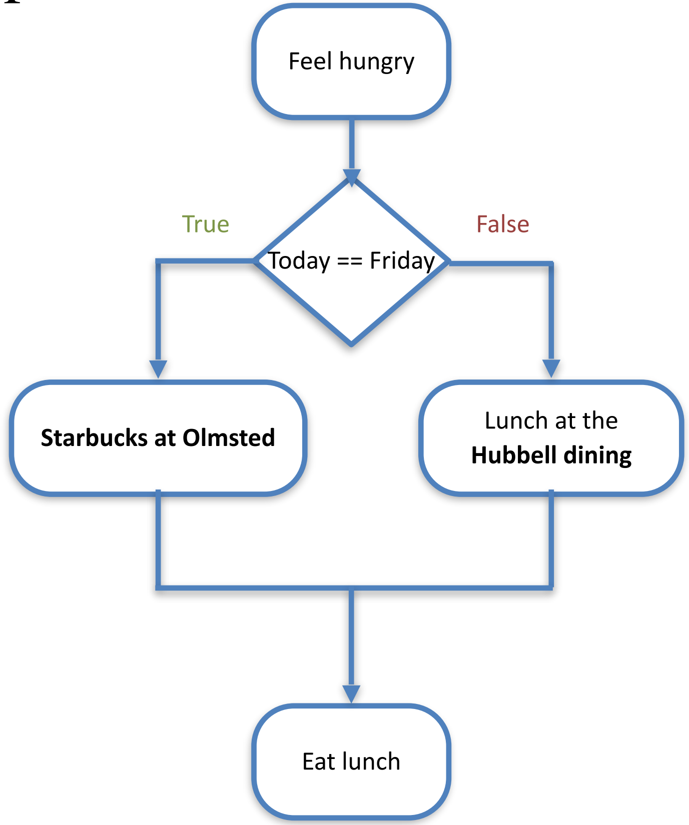
More on function calls
Lab 3 (Task 5)



```python
from graphics import *

def draw_one_rectangle(x1, y1, x2, y2, color, window):

    # your code here
    # ...
    # end of your code
    return None


def build_staircase():

    window = GraphWin("Staircase", 400, 400)
    window.setCoords(0, 0, 400, 400)


    # your code here
    # ...
    side_length_of_rect = 100
    x1 = 0
    y1 = 0
    x2 = 100
    y2 = 100
    draw_one_rectangle(x1, y1, x2, y2, "blue", window)

    # alternatively
    # draw_one_rectangle(0, 0, 100, 100, "blue", window)
    # ...
    # ...
    # end of your code

    return None

build_staircase()
```

Motivation

- Lunch on campus



Topics

- Booleans – new data type
- Expressions with logical operators, comparison operators, and others
- Selection statements
    - if
    - if-else
    - if-elif-…-else

‘**bool**’ Data Type

- Notion of something being true and being false can be represented with two **‘bool’** types in Python
  - True
  - False
- In real life, we always encounter question with Yes or No answer
- Allows us to evaluate true or false questions

Boolean Expression

- Expressions that are evaluates to two ‘**bool**’ types
- Operations with logical operators — and/or/not
  - and – given two boolean, are both True? answer is True

    boolean $expression_1$ and boolean $expression_2$
  - or – given two booleans, at least one is True? answer is True

    boolean $expression_1$ or boolean $expression_2$
  - not – given a boolean expression, switch between True/False

    not boolean expression

Logical Operators

x	y	x and y
False	False	False
False	True	False
True	False	False
True	True	True

- expression$_1$ and expression$_2$

x	y	x or y
False	False	False
False	True	True
True	False	True
True	True	True

expression$_1$ or expression$_2$

x	not x
False	True
True	False

not expression

Comparison Operators

- We can write expression that evaluates to boolean with other comparison operators
  - Compare two values or check something

Description	Example	Result
Less than	2 < 15	True
Greater than	2 > 15	False
Less than or equal	2 <= 15	True
Greater than or equal	2 >=15	False
Equality check	2 == 15	False
Inequality check	2 != 15	True

More Boolean Expressions

X	Y	X and Y
2 < 15	2 >=15	False
3 < 15	2 ==15	False
3 < 15	15 == 15	True
16 > 15	2 != 15	True

- $expression_1$ and $expression_2$

Poll 1

- Follow the link below and submit your answer

https://bit.ly/3ktA7cy

- expression$_1$ or expression$_2$

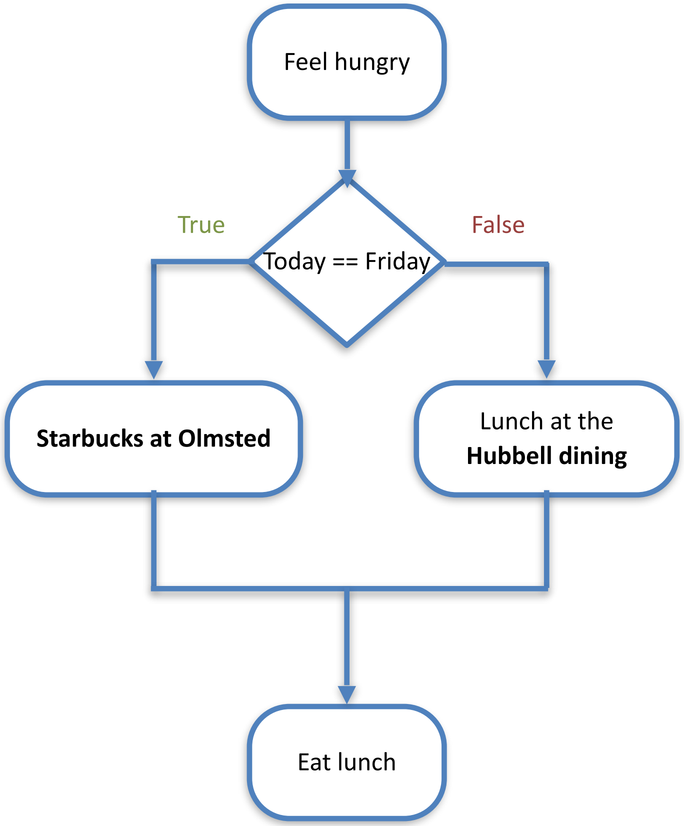
reference

Topics

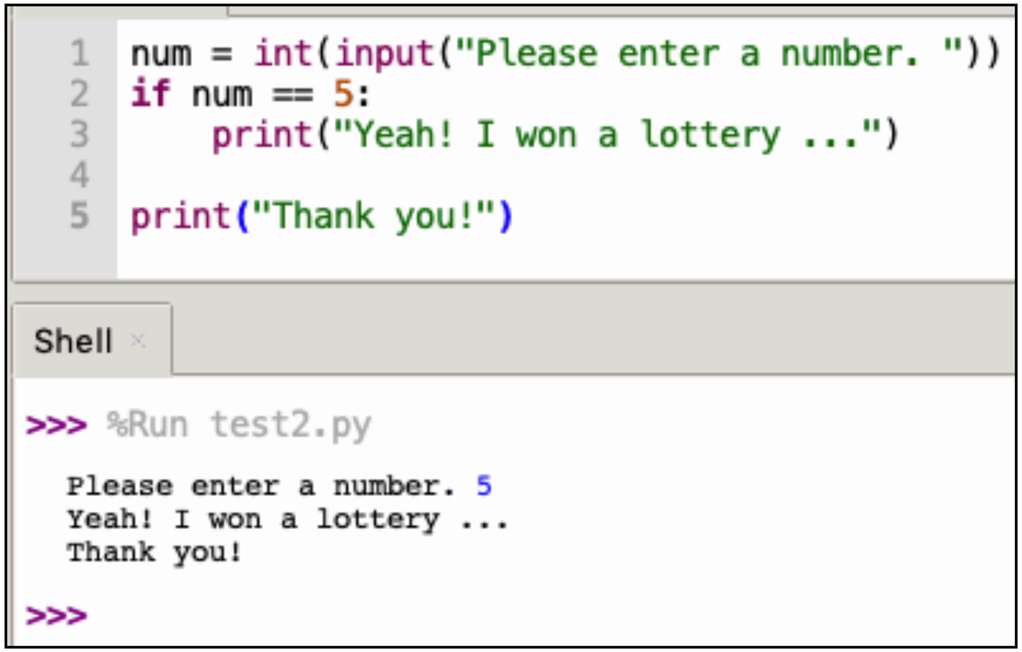
- Booleans – new data type
- Expressions with logical operators, comparison operators, and others
- Selection statements
  - if
  - if-else
  - if-elif-else
    - multiple selection statements

Selection Statements

- program taking one *path* or *branch* of the code instead of taking another, based on the **boolean expression**'s value
- It allows us to ask true/false questions in our code. Depending on the boolean answer (True or False), the program will execute a specific branch.



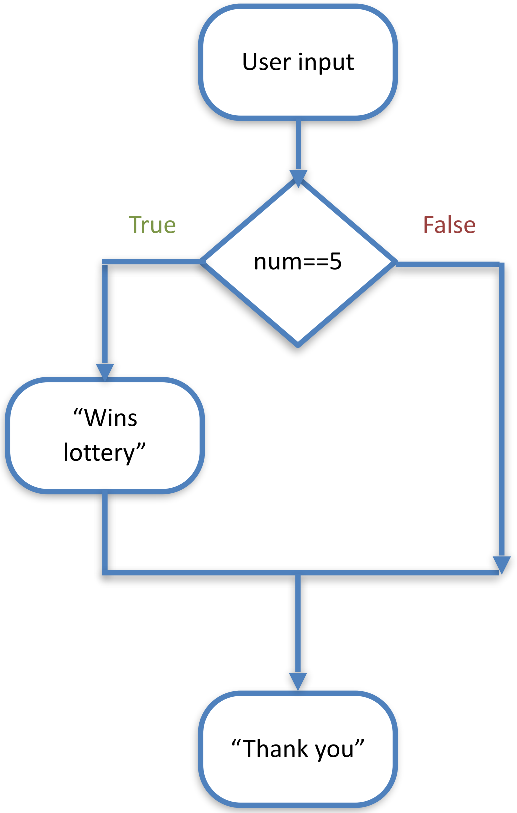
‘if’ Statement



```
num = int(input("Please enter a number. "))
if num == 5:
    print("Yeah! I won a lottery ...")

print("Thank you!")
```

Shell

```
>>> %Run test2.py

  Please enter a number. 5
  Yeah! I won a lottery ...
  Thank you!

>>>
```



‘if’ Statement

- if **<condition expression>** :

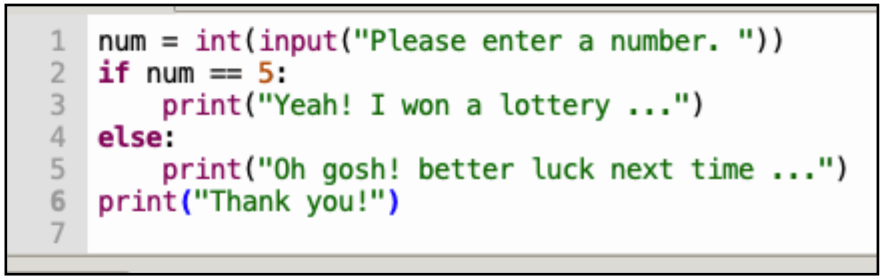
  <block statements>

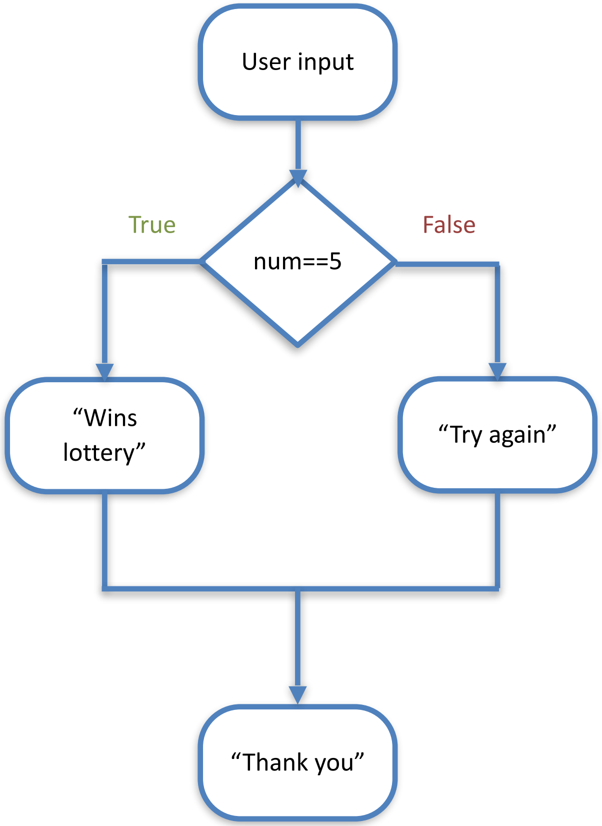
- **condition expression**: a boolean expression

- **block statements**: statements to be executed if result of the condition expression is **True**

- Notice: indention is required to define a **block statements** and also notice a colon at the end of the condition expression

‘if … else’ Statement


```
num = int(input("Please enter a number. "))
if num == 5:
    print("Yeah! I won a lottery ...")
else:
    print("Oh gosh! better luck next time ...")
print("Thank you!")
```



Poll 2

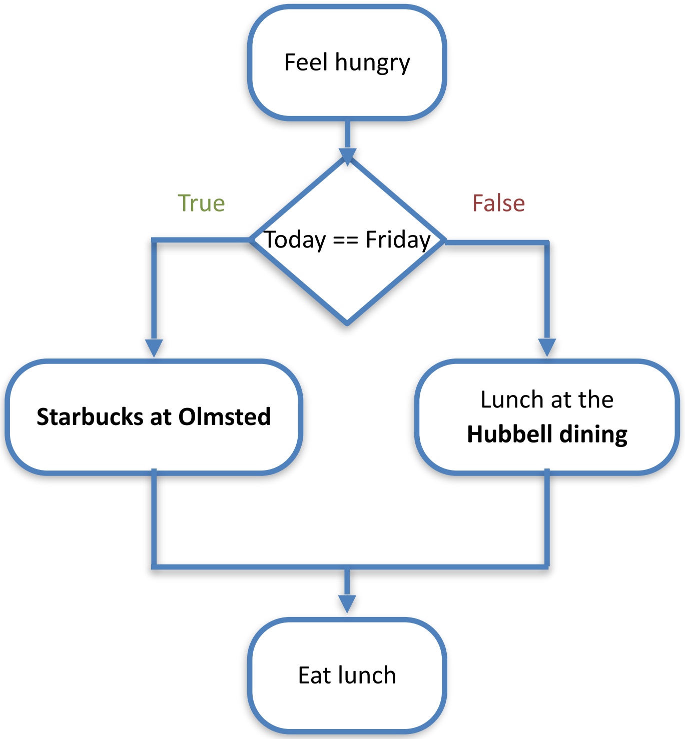
- Follow the link below and submit your answer

https://bit.ly/3nW6534

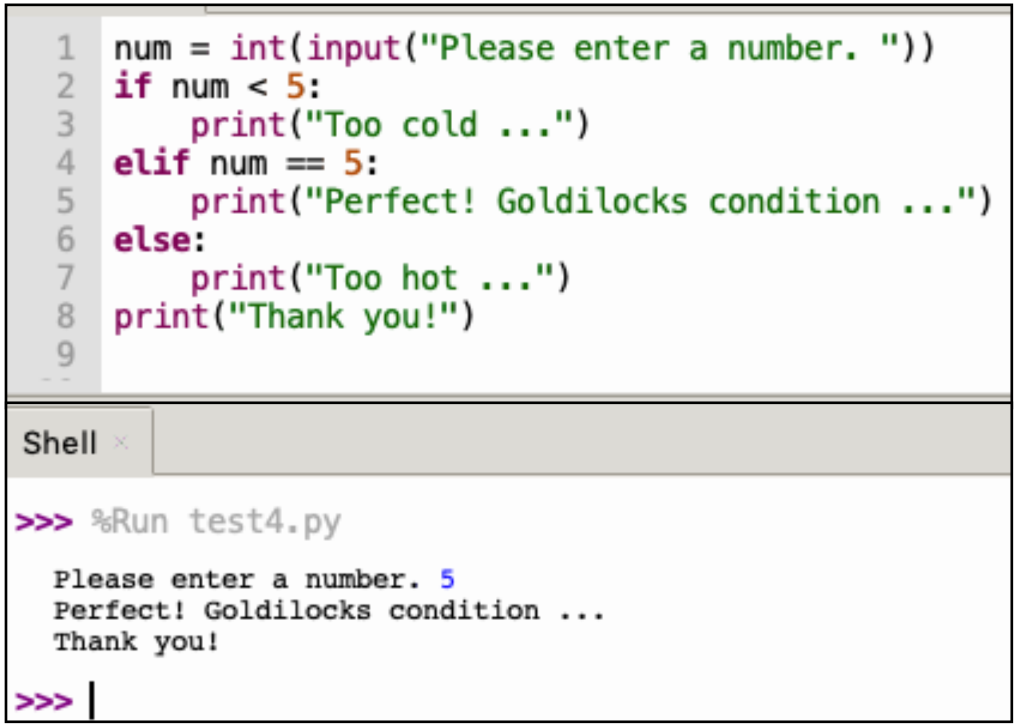
Multiple Selections

- We may need to branch in more than two directions — multiple selection

  - Can have nested if-statement
  - Keyword **elif** (short for 'else if') introduces a new structure
  - Blocks with multiple **elif** conditions structures are referred to as mutually exclusive structures.



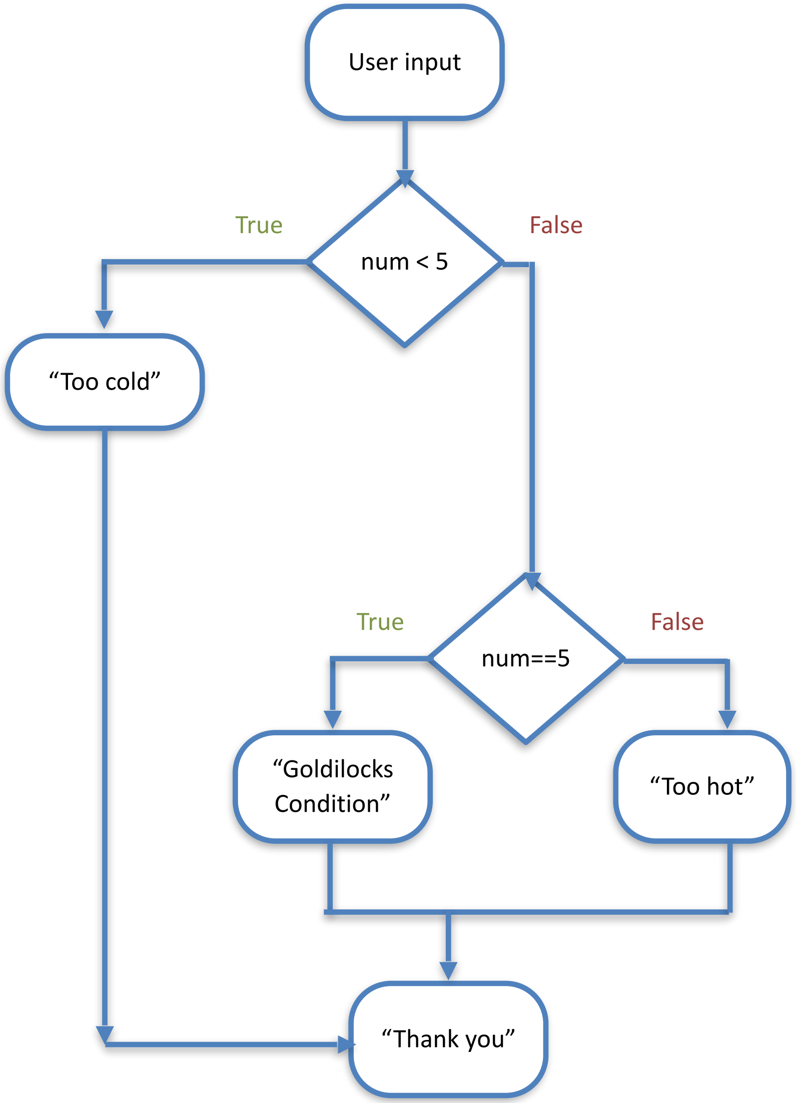
Multiple Selections with else



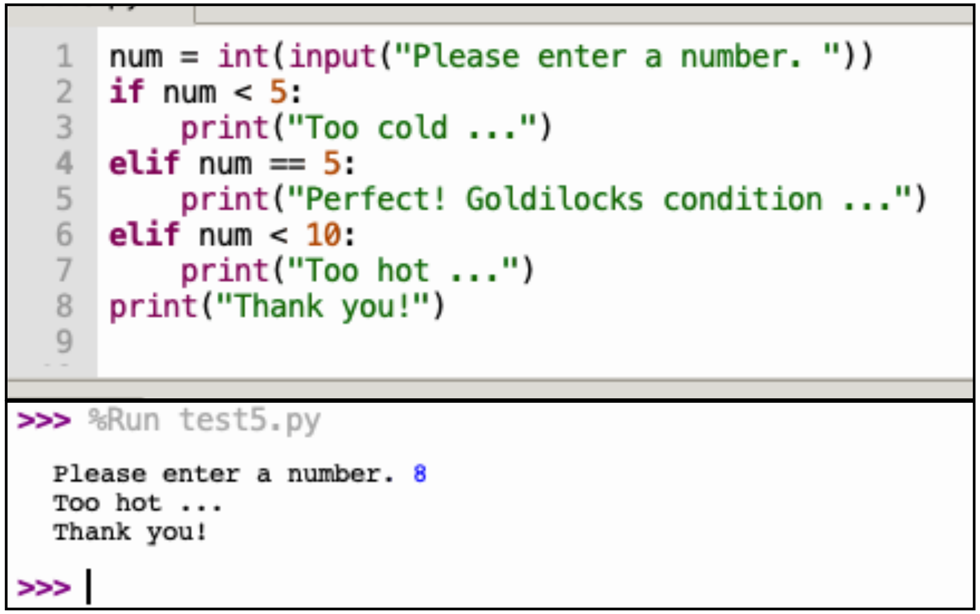
```python
num = int(input("Please enter a number. "))
if num < 5:
    print("Too cold ...")
elif num == 5:
    print("Perfect! Goldilocks condition ...")
else:
    print("Too hot ...")
print("Thank you!")
```

Shell

```
>>> %Run test4.py

  Please enter a number. 5
  Perfect! Goldilocks condition ...
  Thank you!

>>> 
```



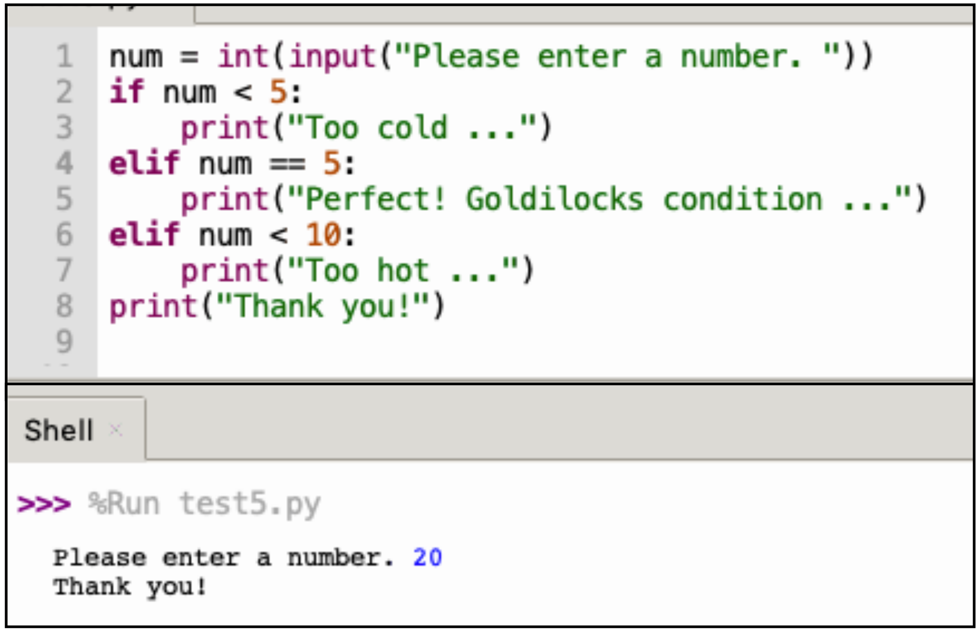
Multiple Selection without 'else'



```python
num = int(input("Please enter a number. "))
if num < 5:
    print("Too cold ...")
elif num == 5:
    print("Perfect! Goldilocks condition ...")
elif num < 10:
    print("Too hot ...")
print("Thank you!")
```

```
>>> %Run test5.py

  Please enter a number. 8
  Too hot ...
  Thank you!

>>> 
```

Multiple Selection without ‘else’



```python
num = int(input("Please enter a number. "))
if num < 5:
    print("Too cold ...")
elif num == 5:
    print("Perfect! Goldilocks condition ...")
elif num < 10:
    print("Too hot ...")
print("Thank you!")
```

Shell

```
>>> %Run test5.py

  Please enter a number. 20
  Thank you!
```

Demo: multiple selections - another example

Practice Problem

Red

Green

Blue

Practice Problem

Ask the user to enter a string.

If user enters “Red” then print(“I like red channel of an RGB image.”)

If user enters “Green” then print(“I like green channel of an RGB image.”)

If user enters “Blue” then print(“I like blue channel of an RGB image.”)

If user enters anything else then print(“Image format is not correct.”)

Summary

- **Takeaway from this lecture**
  - Encountering question with **yes** or **no** answer can be expressed with Python **Boolean** datatype, which has **true** or **false** values.
    - Boolean expression with logical operator (and, or, not)
    - Boolean expression with comparison operator (<, <=, >, ==, etc)
  - Selection statements are useful for branching inside your program
    - if block
    - if-else block
    - if - elif - elif - … - else block
- **Announcements:**
  - Quiz 1, Lab 1 grades are available on Blackboard
  - Next week, there will be another Quiz on *boolean expression* and *selection statements*. Implicitly, there will be function calls — so review functions again!