SELECTIONS

INTERNATIONAL COLLEGE, KMITL

PRESSESIONAL COURSE

OVERALL

- Boolean Types, Values and Expressions
- Generating Random Number
- If statement
- Two Way If – Else Statement
- Nested If and Multi-way if-elif-else Statements
- Logical operators
- Conditional Expression
- Operator Precedence and Associativity
- Common Error in Selection Statement
- Detecting the Location of an Object

BOOLEAN TYPES, VALUES AND EXPRESSIONS

COMPARISON OPERATORS

Python Operator	*Mathematics Symbol*	*Name*	*Example* (radius is 5)	*Result*
`<`	$<$	less than	`radius < 0`	`False`
`<=`	$\leq$	less than or equal to	`radius <= 0`	`False`
`>`	$>$	greater than	`radius > 0`	`True`
`>=`	$\geq$	greater than or equal to	`radius >= 0`	`True`
`==`	$=$	equal to	`radius == 0`	`False`
`!=`	$\neq$	not equal to	`radius != 0`	`True`

The result of the comparison is a **Boolean Value:** True or False

BOOLEAN EXPRESSION :: BOOL()

```
>>> isStudent = True
>>> type(isStudent)
<class 'bool'>
>>> print(isStudent)
True
>>> print(int(isStudent))
1
```

```
>>> isStudent = 1
>>> type(isStudent)
<class 'int'>
>>> print(isStudent)
1
>>> print(bool(isStudent))
True
```

```
>>> print( bool(4))
True
```

```
>>> print( bool(-1))
True
```

```
>>> isAwake = False
>>> type(isAwake)
<class 'bool'>
>>> print(isAwake)
False
>>> print(int(isAwake))
0
```

```
>>> isAwake = 0
>>> type(isAwake)
<class 'int'>
>>> print(isAwake)
0
>>> print(bool(isAwake))
False
```

GENERATING RANDOM NUMBER

RANDOM NUMBER

- Import random module
- Simple Methods:
  - random()
  - randint(a, b)
  - randrange(a, b[, k])

RANDOM NUMBER :: RANDOM()

Return the next random floating point number in the range [0.0, 1.0)

```
>>> import random
>>> random.random()
0.6134590161710275
>>> random.random()
0.2022099530144177
>>> random.random()
0.7927431441102997
```

RANDOM NUMBER :: RANDINT(A, B)

Return a random integer *N* such that a <= N <= b

```
>>> import random
>>> random.randint(1,10)
6
>>> random.randint(1,10)
8
>>> random.randint(1,10)
3
```

RANDOM NUMBER :: RANDRANGE()

randrange(start, stop[, step])

Return a randomly selected element from range(start, stop, step)

```
>>> import random
>>> random.randrange(1,10,3)
4
>>> random.randrange(1,10,3)
7
>>> random.randrange(1,10,3)
1
```

RANDOM NUMBER :: EXAMPLE

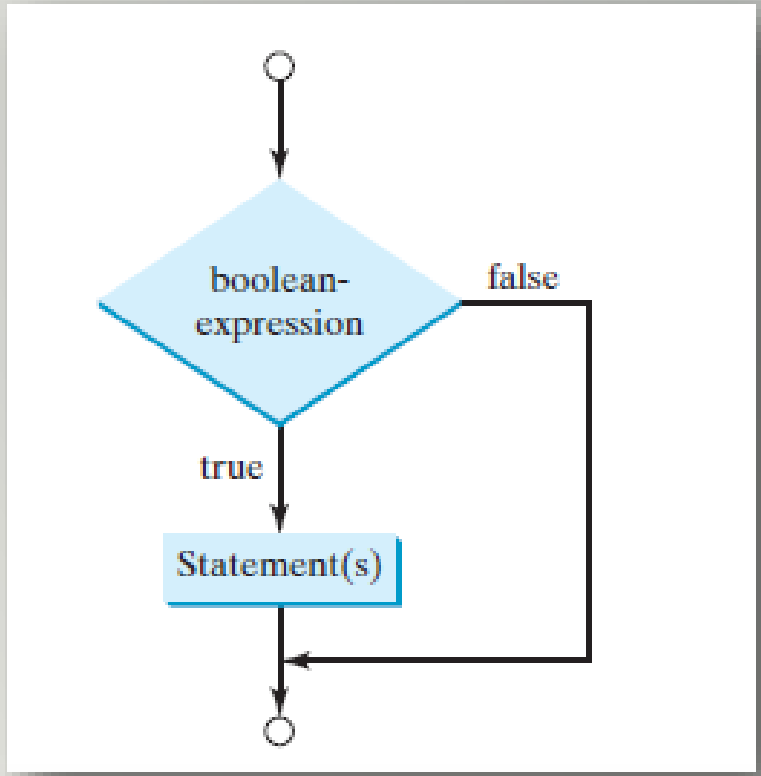
```python
import random

# Generate random numbers
number1 = random.randint(0, 9)
number2 = random.randint(0, 9)

# Prompt the user to enter an answer
answer = eval(input("What is " + str(number1) + " + "
    + str(number2) + "? "))

# Display result
print(number1, "+", number2, "=", answer,
    "is", number1 + number2 == answer)
```

```
What is 1 + 7? 8 ↵Enter
1 + 7 = 8 is True
```

```
What is 4 + 8? 9 ↵Enter
4 + 8 = 9 is False
```

IF STATEMENT

IF STATEMENT :: ONE WAY IF STATEMENT



A one-way **if** *statement executes the statements if the condition is true.*

Syntax:

```
if boolean-expression:
    statement(s)  # Note that the statement(s) must be indented
```

Example:

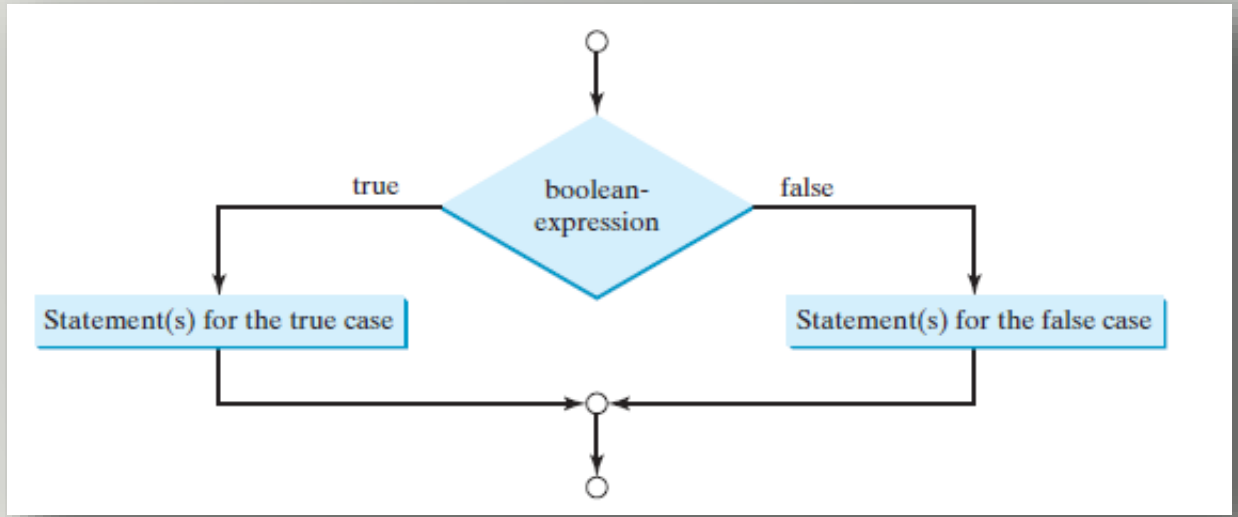
```
number = eval(input("Enter an integer: "))

if number % 5 == 0:
    print("HiFive")

if number % 2 == 0:
    print("HiEven")
```

```
Enter an integer: 4 ↵Enter
HiEven
```

```
Enter an integer: 30 ↵Enter
HiFive
HiEven
```

TWO WAY IF – ELSE STATEMENT

IF – ELSE STATEMENT



A two-way `if-else` statement decides which statements to execute based on whether the condition is true or false.

Syntax:

```
if boolean-expression:
    statement(s)-for-the-true-case
else:
    statement(s)-for-the-false-case
```

IF – ELSE STATEMENT :: EXAMPLES

```
if number % 2 == 0:
    print(number, "is even.")
else:
    print(number, "is odd.")
```

```
if radius >= 0:
    area = radius * radius * math.pi
    print("The area for the circle of radius", radius, "is", area)
else:
    print("Negative input")
```

NESTED IF AND MULTI-WAY IF-ELIF-ELSE STATEMENTS

IF-ELIF-ELSE STATEMENT

```
if score >= 90.0:
    grade = 'A'
else:
    if score >= 80.0:
        grade = 'B'
  else:
      if score >= 70.0:
          grade = 'C'
      else:
          if score >= 60.0:
              grade = 'D'
          else:
              grade = 'F'
```

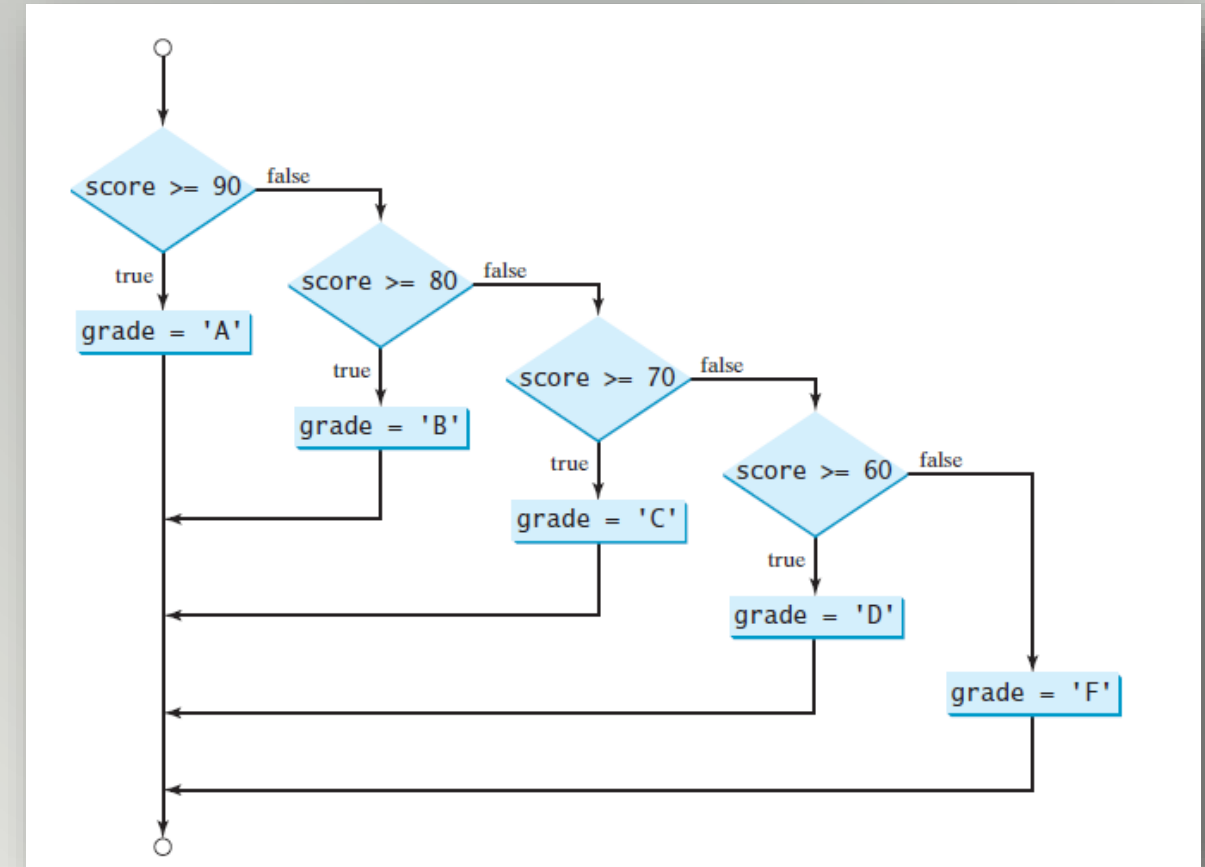
Equivalent

This is better

```
if score >= 90.0:
    grade = 'A'
elif score >= 80.0:
    grade = 'B'
elif score >= 70.0:
    grade = 'C'
elif score >= 60.0:
    grade = 'D'
else:
    grade = 'F'
```

One `if` statement can be placed inside another `if` statement to form a nested `if` statement.

IF-ELIF-ELSE STATEMENT



LOGICAL OPERATORS

LOGICAL OPERATORS :: AND, OR , NOT

TABLE 4.3 Boolean Operators

Operator	*Description*
`not`	logical negation
`and`	logical conjunction
`or`	logical disjunction

LOGICAL OPERATORS :: EXAMPLE

```
number = eval(input("Enter an integer: "))

if number % 2 == 0 and number % 3 == 0:
    print(number, "is divisible by 2 and 3")

if number % 2 == 0 or number % 3 == 0:
    print(number, "is divisible by 2 or 3")

if (number % 2 == 0 or number % 3 == 0) and  \
        not (number % 2 == 0 and number % 3 == 0):
    print(number, "is divisible by 2 or 3, but not both")
```

```
Enter an integer: 18
18 is divisible by 2 and 3
18 is divisible by 2 or 3
```

```
Enter an integer: 15
15 is divisible by 2 or 3
15 is divisible by 2 or 3, but not both
```

CONDITIONAL EXPRESSION

CONDITIONAL EXPRESSION

```
if x > 0:
    y = 1
else:
    y = -1
```

Is equivalent to:

```
y = 1 if x > 0 else -1
```

Syntax:

```
expression1 if boolean-expression else expression2
```

OPERATOR PRECEDENCE AND ASSOCIATIVITY

OPERATOR PRECEDENCE CHART

Precedence	*Operator*
↓	+, - (Unary plus and minus)
	** (Exponentiation)
	not
	*, /, //, % (Multiplication, division, integer division, and remainder)
	+, - (Binary addition and subtraction)
	<, <=, >, >= (Comparison)
	==, != (Equality)
	and
	or
	=, +=, -=, *=, /=, //=, %= (Assignment operators)

```
a - b + c - d   ═══ is equivalent to ═══   ((a - b) + c) - d
```

COMMON ERROR IN SELECTION STATEMENT

COMMON ERROR :: INDENT

```
radius = -20

if radius >= 0:
    area = radius * radius * math.pi
print("The area is", area)
```

```
radius = -20

if radius >= 0:
    area = radius * radius * math.pi
    print("The area is", area)
```

COMMON ERROR :: INDENT

```
i = 1
j = 2
k = 3

if i > j:
     if i > k:
          print('A')
else:
    print('B')
```

```
i = 1
j = 2
k = 3

if i > j:
     if i > k:
          print('A')
     else:
          print('B')
```

COMMON ERROR :: TIPS

```
if number % 2 == 0:
    even = True
else:
    even = False
```

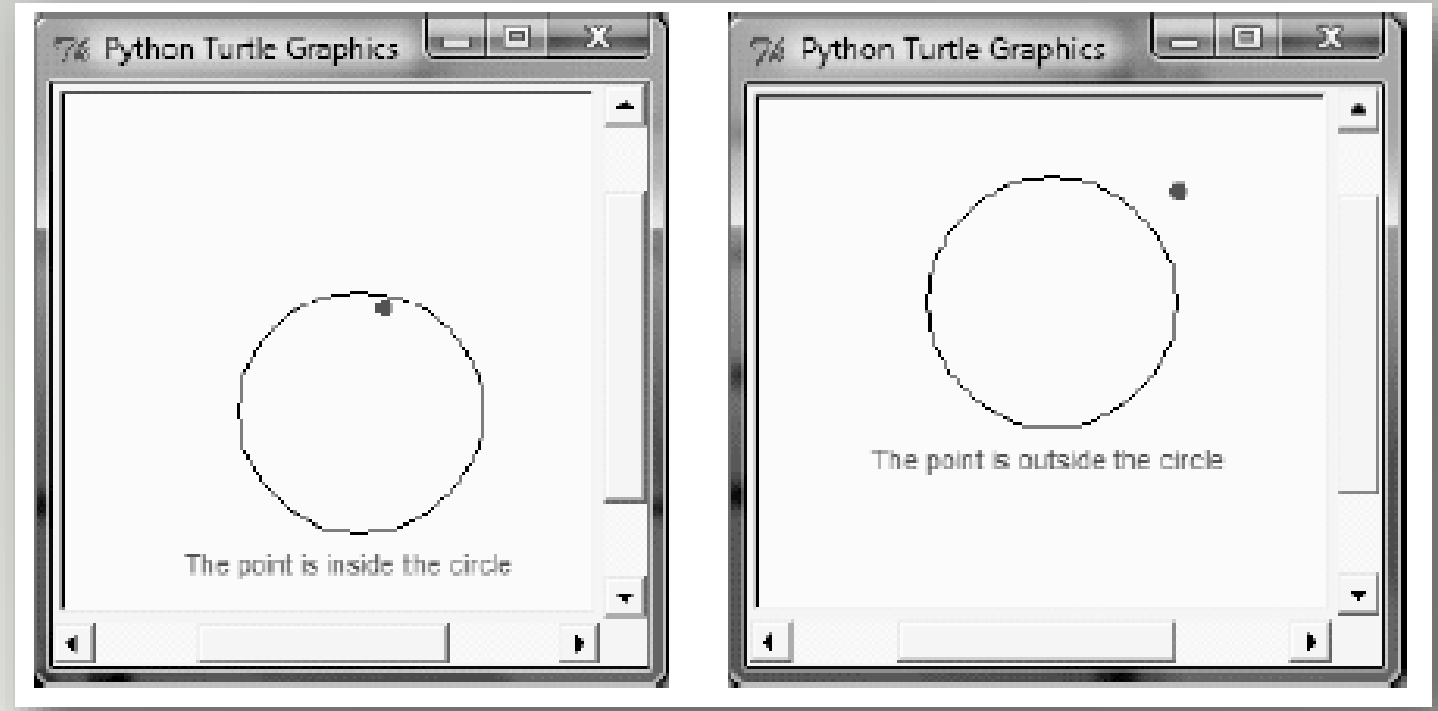
Equivalent

This is shorter

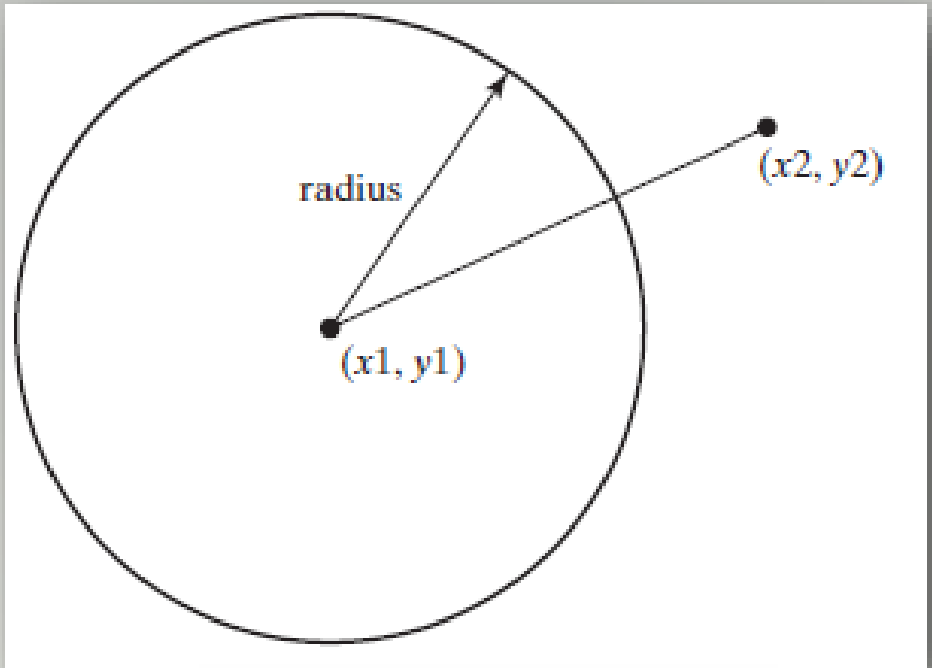
```
even = number % 2 == 0
```

DETECTING THE LOCATION OF AN OBJECT

TURTLE DLO :: TASK



TURTLE DLO :: DISTANCE



Distance Formula

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

TURTLE DLO :: CODE

```
import turtle

x1, y1 = eval(input("Enter the center of a circle x, y: "))
radius = eval(input("Enter the radius of the circle: "))
x2, y2 = eval(input("Enter a point x, y: "))

# Draw the circle
turtle.penup()        # Pull the pen up
turtle.goto(x1, y1 - radius)
turtle.pendown()      # Pull the pen down
turtle.circle(radius)
# Draw the point
```

TURTLE DLO :: CODE

```
turtle.penup()        # Pull the pen up
turtle.goto(x2, y2)
turtle.pendown()      # Pull the pen down
turtle.begin_fill()   # Begin to fill color in a shape
turtle.color("red")
turtle.circle(3)
turtle.end_fill()     # Fill the shape

# Display the status
turtle.penup()        # Pull the pen up
turtle.goto(x1 - 70, y1 - radius - 20)
turtle.pendown()

d = ((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1)) ** 0.5
if d <= radius:
    turtle.write("The point is inside the circle")
else:
    turtle.write("The point is outside the circle")

turtle.hideturtle()

turtle.done()
```