Python语言程序设计

# Python Programming

Session 05



Tom Lotz (tom.lotz@outlook.com)

# Content


| | |
|---|---|
| | Review |
| 01 | Boolean Values and Comparisons |
| 02 | Conditional Statements |
| 03 | Practice and homework |

# Review

# A For Loop with range()




- range(3) provides the numbers 0, 1, and 2.
- number stores one value during each repetition.

```
x = range(3)
print(list(x))

# printed output: [0, 1, 2]


for number in range(3):
    print(number, "Hello")
```

# A For Loop with a String




- A string is a sequence of characters.
- The loop takes one character at a time.
- letter stores the current character.

```
for letter in "cat":
    print(letter)
```

# A For Loop with a List




- The loop now takes values from a list.
- item stores one list item during each repetition.

```
items = ["tea", "fruit cup", "sandwich"]

for item in items:
    print(item)
```

# Boolean Values and Comparisons

# True and False

- A comparison produces a Boolean value: True or False.

Python code

```
print(10 > 5)
print(10 < 5)
print(10 == 5)
```

Expected output

```
True
False
False
```

# Comparison Operators

| Operator | Meaning |
|---|---|
| == | Equal |
| != | Not equal |
| > | Greater than |
| < | Less than |
| >= | Greater than or equal |
| <= | Less than or equal |

= assigns a value.  == compares two values.

# Exercise 1: Predict Each Result

- Write down all five results before running the code.

Python code

```
x = 8
y = 12

print(x > y)
print(x < y)
print(x == 8)
print(y != 12)
print(x >= 8)
```

# Exercise 1: Solution

- >= includes equality. != asks whether values differ.

Expected output

```
False
True
True
False
True
```

# Exercise 2: Write Comparisons

- Given a = 15 and b = 20, write and print four comparisons:
- Is a smaller than b? Does a equal b?
- Is a at least 10? Is b different from 25?

OPTIONAL CHALLENGE

Finish first. Optional: choose values that make each comparison False.

# Exercise 2: Solution

Python code

```
a = 15
b = 20

print(a < b)
print(a == b)
print(a >= 10)
print(b != 25)
```

Expected output

```
True
False
True
True
```

# Boolean Values and Strings

- True is a Boolean value. "True" is a string.
- Quotation marks change the type of value.
- Values can print the same text but have different types.

Python code

```
result = True
print(result)

result = "True"
print(result)

print(True == "True")
```

Expected output

```
True
True
False
```

# Conditional Statements

# The if Statement

- Python checks the condition. True runs the indented block.
- False skips the block. Code outside it continues normally.

Python code

```
x = 10

if x > 5:
    print("Large")

print("Finished")
```

# A Variable Can Store a Boolean

- x > 5 produces a Boolean value and stores it in result.
- if checks the value stored in result.

Python code

```
x = 10
result = x > 5

print(result)

if result:
    print("Condition is true")
```

# Practice: A Boolean Condition

- Predict every output before running the code.

Python code

```
x = 3
result = x > 5

print(result)

if result:
    print("Condition is true")
```

# Boolean Condition: Solution

- 3 > 5 is False. result stores False.
- print(result) still runs. Python skips the indented print().

Expected output

```
False
```

# Exercise 3: Predict the Output

- What prints when x is 3? What changes when x is 8?

Python code

```
x = 3

if x > 5:
    print("A")

print("B")
```

# Exercise 3: Solution

- x = 3: the condition is False, so Python skips print("A").
- x = 8: the condition is True, so Python runs both prints.

x = 3

```
B
```

x = 8

```
A
B
```

# Exercise 4: Find the Errors

- This code contains syntax errors. Identify and fix them.

Code to repair

```
x = 10

if x = 10
print("Equal")
```

# Exercise 4: Solution

- Compare with ==, add the colon, and indent print().

Python code

```
x = 10

if x == 10:
    print("Equal")
```

# Exercise 5: Write an if Statement

- Start with number = 25.
- Print "Greater than 20" only when number is greater than 20.
- Test your program with 25 and 15.

OPTIONAL CHALLENGE

Finish first. Optional: add a second message that always prints.

# Exercise 5: Solution

- With 25: Greater than 20. With 15: no output.

Python code

```
number = 25

if number > 20:
    print("Greater than 20")
```

# Two Possible Outcomes

- True runs the if block. False runs the else block.
- Exactly one of these branches runs.
- else has a colon, but no condition.

Python code

```
x = 10

if x > 5:
    print("Large")
else:
    print("Small")
```

# Exercise 6: Even or Odd

- % gives the remainder. A remainder of 0 means even.
- Predict and test with number = 7, 12, and 0.

Python code

```
number = 7

if number % 2 == 0:
    print("Even")
else:
    print("Odd")
```

# Exercise 6: Solution

- 7 % 2 is 1, so 7 prints Odd.
- 12 % 2 is 0, so 12 prints Even.
- 0 % 2 is 0, so 0 prints Even.

# Exercise 7: Positive or Otherwise

- Store a number in x.
- If x is greater than zero, display "Positive".
- Otherwise, display "Zero or negative".
- Test with a positive number, zero, and a negative number.

OPTIONAL CHALLENGE

Finish first. Optional: compare two numbers. Report the larger one, or say they are equal.

# Exercise 7: Solution

- Try x = 4, x = 0, and x = -3.

Python code

```
x = 4

if x > 0:
    print("Positive")
else:
    print("Zero or negative")
```

# Practice: Repair a Number Test

- Repair this program so it classifies integer values correctly.
- Above 5: "Greater than 5". Otherwise: "5 or below".
- Test with 4, 5, and 12.

Code to repair

```
x = input("Number: ")

if x == "5":
    print("Greater than 5")
else:
    print("5 or below")
```

# Number Test: Solution

- Convert the input to an integer. Compare it with the number 5.
- The program prints 5 or below for both 4 and 5.

Python code

```
x = int(input("Number: "))

if x > 5:
    print("Greater than 5")
else:
    print("5 or below")
```

# More Than Two Outcomes

- Python checks conditions from top to bottom.
- elif tests another condition if the earlier ones were False.
- After a match, Python skips the remaining branches.

Python code

```
x = 0

if x > 0:
    print("Positive")
elif x < 0:
    print("Negative")
else:
    print("Zero")
```

# Execution Order

Check x > 0 → True → print("Positive")

↓ False

Check x < 0 → True → print("Negative")

↓ False

else: print("Zero")

Every selected branch continues after the chain.

# Exercise 8: Which Branch Runs?

- Predict the output for x = 25, 15, 5, and 20.
- Pay attention to the boundary value 20.

Python code

```
x = 15

if x > 20:
    print("A")
elif x > 10:
    print("B")
else:
    print("C")
```

# Exercise 8: Solution

- 25 prints A. 15 prints B. 5 prints C. 20 prints B.
- 20 > 20 is False, but 20 > 10 is True.
- If x > 10 comes first, 25 matches it too.
- Put the conditions in the order your program needs.

# Exercise 9: if or elif?

- Predict both programs for x = 15, x = 5, and x = -5.

Program A

```
x = 15

if x > 10:
    print("Greater than 10")

if x > 0:
    print("Positive")
```

Program B

```
x = 15

if x > 10:
    print("Greater than 10")
elif x > 0:
    print("Positive")
```

# Exercise 9: Solution

Program A: output

```
x = 15
Greater than 10
Positive

x = 5
Positive

x = -5
(no output)
```

Program B: output

```
x = 15
Greater than 10

x = 5
Positive

x = -5
(no output)
```

- Independent if statements each check their condition. A chain selects at most one branch.
- Order matters: if x > 0 came first, a later elif x > 10 would never run.

# Exercise 10: Classify a Number

- Less than 0: print "Negative".
- From 0 to 10 inclusive: print "Small".
- Greater than 10: print "Large".
- Test with -5, 0, 10, and 20.

# Exercise 10: Solution

- After x < 0 is False, x is already 0 or greater.

Python code

```
x = -5

if x < 0:
    print("Negative")
elif x <= 10:
    print("Small")
else:
    print("Large")
```

# Practice

# Independent Practice — Simple Calculator

## CORE EXERCISE

- Ask for two integer numbers.
- Ask for +, -, or *.
- Use if, elif, and else to select the calculation.
- Display the result.
- Show "Unknown operation" for an unrecognized operation.

Example: 12, 5, * produces 60.

## OPTIONAL CHALLENGES

Finish the core first.
Choose challenges in any order.

A. Add division and prevent division by zero.

B. Accept decimal values instead of only integers.

C. Add exponentiation (**).

D. Add an operation to compare the numbers: report the larger value, or say they are equal.

# Calculator Solution

Python code

```
a = int(input("First number: "))
b = int(input("Second number: "))
operation = input("Operation (+, -, *): ")

if operation == "+":
    print("Result:", a + b)
elif operation == "-":
    print("Result:", a - b)
elif operation == "*":
    print("Result:", a * b)
else:
    print("Unknown operation")
```

# Understanding the Calculator

- int(input()) converts the entered text into an integer.
- input() stores the operation as a string.
- "+" is a string representing the selected operation.
- a + b adds two numerical values.
- Only one branch of the conditional chain runs.

# Final Check

1. Predict the output of this program.
2. Explain the difference between = and ==.
3. Write an if statement that prints "Correct" when answer equals 5.

Python code

```
x = 10

if x > 10:
    print("A")
elif x == 10:
    print("B")
else:
    print("C")
```

# What You Learned Today

- Comparisons produce Boolean values: True or False.
- = assigns a value. == compares values.
- if, elif, and else choose which branch runs.
- Colons and indentation define the blocks.
- Separate if statements each check their own condition.
- A conditional chain selects at most one branch.
- int(input()) reads an integer. input() reads text.