GET214:
COMPUTING AND SOFTWARE ENGINEERING

Lesson 3

DECISIONS

Learning outcomes

At the end of the lesson, students should be able:

1. Explain a Boolean value
2. Use bool variables to store Boolean values
3. Demonstrate converting integers, floats, and strings to Booleans
4. Demonstrate converting Booleans to integers, floats, and strings
5. Use comparison operators to compare integers, floats, and strings
6. Create an if-else statement to perform an operation when a condition is true and another operation otherwise
7. Create expressions with logical operators
8. Create a chained decision statement to evaluate multiple conditions.
9. Implement a program with nested if-else statements
10. Create a conditional expression

Introduction to Decisions

- Python interpreter allows a flow of execution from top to buttom under normal circumstances.
- A programmer may chose to create multiple branches and run each based on certain conditions
- A **branch** is a group of statements that execute based on a condition.
- The condition for a branch to execute is usually evaluated to a Boolean value.

Boolean Values and the bool Data Type

- People often ask binary questions such as yes/no or true/false questions.
- For example, Do you like pineapple on pizza? The answer would be **yes or no/ true or false**: I like pineapple on pizza.
- The response is a Boolean value, meaning the value is either true or false.
- The **bool** data type, standing for Boolean, represents a binary value of either true or false.
- True and False are keywords, and capitalization is required.

Type Conversion with bool()

- Deciding whether a value is true or false is helpful when writing programs/statements based on decisions.
- Converting data types to Booleans can seem unintuitive at first.
- For instance, is "ice cream" True? But the conversion is actually simple.
- **bool()** converts a value to a Boolean value, True or False.
  - True: any non-zero number, any non-empty string
  - False: 0, empty string

Comparison Operators

S/N	Operator	Name	Description
1	==	Equal	Returns **True** if the value of the expression on the right equals the value held by the variable on the left. Otherwise, **False**.
2	!=	Not equal	Returns **True** if the value of the expression on the right does not equal the value held by the variable on the left. Otherwise, **False**.
3	<	Less than	Returns **True** if the value of the variable on the left is less than the value of the expression on the right. Otherwise, **False**.
4	<=	Less or equal	Returns **True** if the value of the variable on the left is less or equal to the value of the expression on the right. Otherwise, **False**.
5	>	Greater than	Returns **True** if the value of the variable on the left is greater than the value of the expression on the right. Otherwise, **False**.
6	>=	Greater or equal	Returns **True** if the value of the variable on the left is greater or equal to the value of the expression on the right. Otherwise, **False**.

Activity 3.1: Even number

- Write a program that reads in an integer and prints whether the integer is even or not.
- Remember, a number is even if the number is divisible by 2.
- To test this use number % 2 == 0.
- Ex: If the input is 6, the output is "6 is even: True".

The If-else Statement

- A **condition** is an expression that evaluates to true or false.
- An **if statement** is a decision-making structure that contains a condition and a body of statements.
- If the condition is true, the body is executed.
- If the condition is false, the body is not executed.
- The if statement's body must be grouped together and have one level of indentation.
- Four spaces per indentation level is recommended.

The else side if the if...else statement

- An **else statement** is used with an if statement and contains a body of statements that is executed when the if statement's condition is false.
- When an if-else statement is executed, one and only one of the branches is taken.
- That is, the body of the if or the body of the else is executed.
- Note: The else statement is at the same level of indentation as the if statement, and the body is indented.

Activity 3.2: Converting Temperature Scales

- The following program reads in a temperature as a float and the unit as a string: "f" for Fahrenheit or "c" for Celsius.
- Calculate new_temp, the result of converting temp from Fahrenheit to Celsius or Celsius to Fahrenheit based on unit. Calculate new_unit: "c" if unit is "f" and "f" if unit is "c".
- **Conversion formulas:**
  - Degrees Celsius = (degrees Fahrenheit - 32) * 5/9
  - Degrees Fahrenheit = (degrees Celsius * 5/9) + 32

Boolean Operations

- Decisions are often based on multiple conditions.
- Eg: A program printing if a business is open may check that hour >= 9 and hour < 17.
- A **logical operator** takes condition operand(s) and produces True or False.
- Python has three logical operators: **and**, **or**, and **not**.
- The **and** operator takes two condition operands and returns True if both conditions are true.

Truth Table for and Operation

p	q	p and q
True	True	True
True	False	False
False	True	False
False	False	False

Truth Table for or Operation

p	q	p or q
True	True	True
True	False	True
False	True	True
False	False	False

Truth table for the not operation

- The **not** operator takes one condition operand and returns True when the operand is false and returns False when the operand is true.
- **not** is a useful operator that can make a condition more readable and can be used to toggle a Boolean's value.

p	not p
True	False
False	True

Activity 3.3: Speed limits

- Write a program that reads in a car's speed as an integer and checks if the car's speed is within the freeway limits. A car's speed must be at least 45 mph but no greater than 70 mph on the freeway.
- If the speed is within the limits, print **"Good driving".** Else, print **"Follow the speed limits"**.

Chained Decision with elif

- Sometimes, a complicated decision is based on more than a single condition.
- For example, a travel planning site reviews the layovers on an itinerary.
- If a layover is greater than 24 hours, the site should suggest accommodations.
- Else if the layover is less than one hour, the site should alert for a possible missed connection.

- Chaining decision statements with elif allows the programmer to check for multiple conditions.
- An **elif** (short for else if) statement checks a condition when the prior decision statement's condition is false.
- An elif statement is part of a chain and must follow an if (or elif) statement.

if-elif statement template:

```
# Statements before
if condition:
    # Body
elif condition:
    # Body
# Statements after
```

Example 3.1: Grade Computation

- Write a program that will take a total score from a user and determine what grade he should get based on the university of Uyo grading scheme.

Nested Decision Statements

- Suppose a programmer is writing a program that reads in a game ID and player count and prints whether the user has the right number of players for the game.

The programmer may start with:

```
if game == 1 and players < 2:
    print("Not enough players")
if game == 1 and players > 4:
    print("Too many players")
if game == 1 and (2 <= players <= 4):
    print("Ready to start")
if game == 2 and players < 3:
    print("Not enough players")
if game == 2 and players > 6:
    print("Too many players")
if game == 2 and (3 <= players <= 6):

    print("Ready to start")
```

- The programmer realizes the code is redundant. What if the programmer could decide the game ID first and then make a decision about players? Nesting allows a decision statement to be inside another decision statement, and is indicated by an indentation level.

An improved program:

```
if game == 1:
    if players < 2:
    print("Not enough players")
    elif players > 4:
    print("Too many players")
    else:
    print("Ready to start")
if game == 2:
    if players < 3:
    print("Not enough players")
    elif players > 6:
    print("Too many players")
    else:
    print("Ready to start")

# Test game IDs 3-end
```

Activity 3.4: Meal Orders

- Write a program that reads in a string, "lunch" or "dinner", representing the menu choice, and an integer, 1, 2, or 3, representing the user's meal choice. The program then prints the user's meal choice.
- Lunch Meal Options
  - 1: Caesar salad
  - 2: Spicy chicken wrap
  - 3: Butternut squash soup
- Dinner Meal Options
  - 1: Baked salmon
  - 2: Turkey burger
  - 3: Mushroom risotto

Example: If the input is:

lunch
3

The output is:

Your order: Butternut squash soup

Conditional Expressions

- A **conditional expression** (also known as a "ternary operator") is a simplified, single-line version of an if-else statement.
- Conditional expression template:

```
expression_if_true if condition else expression_if_false
```

- A conditional expression is evaluated by first checking the condition.
- If condition is true, expression_if_true is evaluated, and the result is the resulting value of the conditional expression.
- Else, expression_if_false is evaluated, and the result is the resulting value of the conditional expression.

A variable can be assigned with a conditional expression. For example, finding the max of two numbers can be calculated with

```
max_num = y if x < y else x
```

The above is equivalent of the following code:

```
if x < y:
    max_num = y
else:
    max_num = x
```

Activity 3.5: Ping values

- Write a program that reads in an integer, ping, and prints ping_report, a string indicating whether the ping is low to average or too high.
- ping values under 150 have a ping_report of "low to average".
- ping values of 150 and higher have a ping_report of "too high".
- Use a conditional expression to assign ping_report.
- Example: If the input is 30, the output is "Ping is low to average"

Questions to attempt from reference text

Concept in Practice Questions

Page 92 - 117