Introduction to Programming

Functions

- **Simple Decision**
- **Two-Way Decision**
- Multi-Way Decision
- Exception Handling
- Study in Design: Max of Three

Decision – Why?

- So far, we've viewed programs as sequences of instructions that are followed one after the other.
- This is a fundamental programming concept
- We want to be able to alter the sequential flow of a program.
  - We have seen an example when we used loop to repeat a set of statements multiple times.

Decision – Why?

- Sometime we need to make a decision to solve a problem.
  - We may want to do certain things under certain condition
  - Input checking
  - Error handling

Decision Making

- Take action based on a condition
- If the condition is true take one action
- If the condition is false take a different action
  - Or no action
- It provides capability to make a decision
- Take alternate course of action based on the outcome of decision
- Control structures allow us to alter sequential program flow.

Decision Making

- Types of decision making
  - Simple decision
  - Two-way decision
  - Multi-way decision

Simple Decision

- We take an action if the condition is true
  - If condition is false we don't do anything
- Example
  - If the sun is shining then I will read book.

Simple Decision

- The Python `if` statement is used to implement the decision.
- `if <condition>:`

```
    body (one or more statements)

next statement
```

- The body is a sequence of one or more statements indented under the `if` heading
- Good idea to leave a blank line after the body of if construct!

Simple Decision

- The semantics of the `if` should be clear.
  - First, the condition in the heading is evaluated.
  - If the condition is true, the sequence of statements in the body is executed, and then control passes to the next statement in the program.
  - If the condition is false, the statements in the body are skipped, and control passes to the next statement in the program.

Forming Simple Conditions

- What does a condition look like?
- `<expr> <relop> <expr>`
- `<relop>` is short for relational operator

Forming Simple Conditions

Python	Mathematics	Meaning
<	<	Less than
<=	≤	Less than or equal to
==	=	Equal to
>=	≥	Greater than or equal to
>	>	Greater than
!=	≠	Not equal to

Forming Simple Conditions

- Conditions may compare either numbers or strings.
- When comparing strings, the ordering is lexicographic, meaning that the strings are sorted based on the underlying Unicode. Because of this, all upper-case Latin letters come before lower-case letters. (“Bbbb” comes before “aaaa”)

Forming Simple Conditions

- Conditions are based on Boolean expressions, named for the English mathematician George Boole.
- When a Boolean expression is evaluated, it produces either a value of true (meaning the condition holds), or it produces false (it does not hold).
- Some computer languages use 1 and 0 to represent “true” and “false”
- Boolean conditions are of type bool and the Boolean values of true and false are represented by the literals True and False.

Simple Decision

- Example
  - Write a Python program that will ask user for their age.
  - If age >= 18 print then the message that they can drive
  - What if age < 18?
  - Not a good solution!
  - We don't know

Two-Way Decisions

- We could add another if to the end:
-
```
if age < 0:
    print("You cannot drive!" )
```
- This works, but feels wrong. We have two decisions, with mutually exclusive outcomes (if `age >= 18` then `age < 18` must be false, and vice versa).

Two-Way Decisions

- We take an action if the condition is true
  - If condition is false then we take a different action
- Example
  - If the sun is shining then I will go to the beach otherwise I will go to class.

Two-Way Decisions

- In Python, a two-way decision can be implemented by attaching an else clause onto an if clause.
- This is called an if-else statement:
-
```
if <condition>:
   <statements>
else:
   <statements>
```

Two-Way Decisions

- When Python encounters this structure, it first evaluates the condition. If the condition is true, the statements under the `if` are executed.
- If the condition is false, the statements under the `else` are executed.
- In either case, the statements following the `if-else` are executed after either set of statements are executed.

Class Work

1)Many companies pay time-and-a-half for any hours worked above 40 in a given week. Write a Python program to input the number of hours worked and hourly rate and calculate the total wages for the week.

2)What is wrong with the following code fragment?

```
if (length = min_length):
    print ('The length is minimal')
```

Class Work

1)What output is produced by the following fragment:

```
num = 87
max = 25
if (num >= max * 2):
    print (“apple”)
print (“orange”)
print (“pear”)
```