Fundamentals of Python First Programs

Chapter 1 Exercises

**1. List three common types of computing agents**.

a. CPU

b. Memory

c. Inputs and Outputs

**2.** **Write an algorithm that describes the second part of the process of making change (counting out the coins and bills).**

START

Step 1: Take the whole amount and take variables for quarter, dime, nickel and penny and initialize them to zero.

Step 2: if wholeamount>=25 divide it by 25 and add 1 to quarter. Repeat until it is false.

Step 3: if wholeamount>=10 divide it by 10 and add 1 to dime. Repeat it until false.

Step 4: if wholeamount>=5 divide it by 5 and add 1 to nickel. Repeat it until false.

Step 5: add wholeamount to penny.

Step 6: No of quarter, dime, nickel and penny shows the changes

END

**3.** **Write an algorithm that describes a common task, such as baking a cake or operating a DVD player**.

Baking a Cake:

START

step 1: Heat oven to 325.F

step 2: Gather the ingredients

step3: Mix the ingredients carefully in the bowl.

step 4: Pour the mixture into the baking pan.

step 5: Bake in the oven for 50 mins

step 6: Repeat bake 5 for minutes until cake top springs back when touched in the center.

step 7: Cool on the rack before cutting.

END

**4.** **Describe an instruction that is not well defined and thus could not be included as a step in an algorithm. Give an example of such an instruction**.

On the other hand, a step that says “divide a number by 0” is not well defined, because no computing

agent could carry it out.

**5.** **In what sense is a laptop computer a general-purpose problem-solving machine**?

Computers can be designed to run a small set of algorithms for performing specialized tasks

such as operating a microwave oven. But we can also build computers, like the one on your

desktop, that are capable of performing a task described by any algorithm. These computers

are truly general-purpose problem-solving machines. They are unlike any machines we have

ever built before, and they have formed the basis of the completely new world in which we live.

**6.** **List four devices that use computers and describe the information that they process. (*Hint:* Think of the inputs and outputs of the devices.)**

Input

Microphone: input voice into the computer

Camera: inputs video into the computer

Output

Monitor: shows images of certain things you request from the computer

Speakers: Output sounds from the computer

**1.** **List two examples of input devices and two examples of output devices**.

Input

1. keyboard

2. Mouse

Output

1. Monitor

2. Speakers (my favorite)

**2.** **What does the central processing unit (CPU) do**?

The part of a computer that is responsible for processing data is the central processing unit (CPU). This device, which is also sometimes called a **processor**, consists of electronic switches arranged to perform simple logical, arithmetic, and control operations. The CPU executes an algorithm by fetching its binary instructions from memory, decoding them, and executing them. Executing an instruction might involve fetching other binary information—the data—from memory as well.

**3.** **How is information represented in hardware memory**?

Computer memory is set up to represent and store information in electronic form. Specifically, information is stored as patterns of **binary digits** (1s and 0s).

**4.** **What is the difference between a terminal-based interface and a graphical user interface**?

A **terminal-based interface** accepts inputs from a keyboard and displays text output on a monitor screen. A **graphical user interface (GUI)** organizes the monitor screen around the metaphor of a desktop, with windows containing icons for folders, files, and applications.

**5.** **What role do translators play in the programming process**?

The programmer then runs another program called a **translator** to convert the high-level program code into executable code. Because it is possible for a programmer to make grammatical mistakes even when writing high-level code, the translator checks for **syntax errors** before it completes the translation process. If it detects any of these errors, the translator alerts the programmer via error messages. The programmer then has to revise the program. If the translation process succeeds without a syntax error, the program can be executed by the **run-time system**.

**1.** **Describe what happens when the programmer enters the string "Greetings!" in the Python shell**.

‘Greetings!’

**2.** **Write a line of code that prompts the user for his or her name and saves the user’s input in a variable called name.**

>>> name = "Martin Flores"

>>> "Hi there, " + name

'Hi there, Martin Flores'

>>> print ('Hi there')

Hi there

>>> print ("Hi there,", name)

Hi there, Martin Flores

>>>

![A screenshot of a cell phone

Description automatically generated]()

**3.** **Answer the question, What is a Python script**?

Script files are also the means by which Python programs are distributed to others. Most important, as you know from writing term papers, files allow you to save, safely and permanently, many hours of work

**4. *Explain what goes on behind the scenes when your computer runs a Python program***.

The command in Step 4 reads the code from the saved file and executes it. If Python executes any print functions in the code, you will see the outputs as usual in the shell window. If the code requests any inputs, the interpreter will pause to allow you to enter them. Otherwise, program execution continues invisibly behind the scenes. When the interpreter has finished executing the last instruction, it quits and returns you to the shell prompt.

**1.** **Suppose your script attempts to print the value of a variable that has not yet been assigned a value. How does the Python interpreter react**?

Python responds that this name is not defined. Although the programmer might have *meant* to write the variable length, Python can read only what the programmer *actually entered*. This is a good example of the rule that a computer can read *only* the instructions it receives, not the instructions we intend to give it.

**2.** **Miranda has forgotten to complete an arithmetic expression before the end of a line of code. How will the Python interpreter react**?

In our final example, the programmer attempts to add two numbers, but forgets to include the second one:

>>> 3 +

SyntaxError: invalid syntax

**3.** **Why does Python code generate fewer types of syntax errors than code in other programming languages**?

There are fewer types of syntax errors to encounter and correct, and a lot less syntax for you to learn!

In our final example, the programmer attempts to add two numbers, but forgets to include the second one:

>>> 3 +

SyntaxError: invalid syntax