Programming – Variables and Data

When we write a computer program we almost always want to store data in some way. By storing data we can change it and use it in different ways.

Data is stored by using **variables**.

1 Using Variables:

A **variable** is a named area of memory which is set aside to store a value. The name given to the variable is called the **variable identifier**. This is a unique name which we use to identify the variable.

Variable identifiers need to be easy for human beings to use. Always try and use sensible names that will make sense.

Exercise 1.1

a) A variable has an identifier userName. What do you think the variable is used for?

b) A variable is needed to store the price of an item in a shop. Which of the following do you think would make a sensible variable identifier:

(i) value
(ii) variable
(iii) price
(iv) item
(v) itemPrice

c) Suggest a suitable identifier for a variable that is used to store the total price of the items in a shopping basket.

Computer programmers often use a style of variable naming called **camel case**. This starts each individual word in a variable identifier with a capital letter except the first word. Examples of variable identifiers using camel case include:

- userName
- dateOfBirth
- itemPrice

Camel case is used because it lets computer programmers look at a program an immediately identify the variables. By using the same convention as other programmers it means that your programs will be easier to read and understand.

The first letter of a variable name is not usually capitalised. Again, this is done to make things easier for other programmers to understand.

Variable identifiers cannot contain spaces. Computers can't look at a phrase such as date of birth and realise that it means one thing. They would see three separate words.

✎ Exercise 1.2

a) Convert the following to suitable variable identifiers using camel case

(i) last name
(ii) address line two
(iii) total price
(iv) phone number

b) Explain why it would be a bad idea to use the variable identifier TotalPrice

---

Identifiers must start with a letter. They cannot start with a number or a symbol.

Identifiers can not use words that have a meaning in a programming language. In Python words such as `print` and `for` have specific meanings. These cannot be used as variable identifiers.

✎ Exercise 1.3

Which of the following would be unsuitable variable identifiers if you were programming in Python

(i) input
(ii) while
(iii) 4name
(iv) price

---

2 Assigning values to variables

When a variable is first used it is **initialised**. This allocates the identifier to the variable and tells the computer to set up the named area of memory required to store values.

A value can then be **assigned** to the variable. This stores a value in the memory location associated with the variable identifier.

When programming with Python, variables are almost always initialised and have values assigned at the same time.

In **pseudo-code** values are assigned to variables using an arrow ←:

```
userName ← "20jsmi01"
```

This assigns the value "20jsmit01" to the variable userName and stores this value in the area of memory which is identified by the variable identifier.

Other examples of variable assignment include:

```
userPassword ← "banana"

highScore ← 340

addressOne ← "22 Acacia Avenue"
```

In each case the value to the **right** is assigned to the variable identified on the **left**.

You need to be able to understand **pseudo-code**. It will be used in exams. You can write any form of code you want to in an exam, but the exam board will always write code using pseudo-code. If you don't understand what the pseudo-code means you won't be able to answer questions.

Exercise 2.1

Write a line of pseudo-code to assign the value 27 to the variable myAge

---

Once a variable has been initialised it can be changed as many times as needed. The value stored by the variable is simply updated each time a new value is assigned.

```
1     myAge ← 27
2     OUTPUT myAge
3     myAge ← 32
4     OUTPUT myAge
```

In this code the variable myAge is initially given the value 27. This is outputted to the screen in line 2. The value is then changed and the new value outputted in line 4.

The pseudo-code command **OUTPUT** simply outputs a value to the screen. In Python the command **`print()`** is used to do the same thing.

✎ Exercise 2.2

The variables myVar and yourVar are used in this pseudo-code. Write down the values of both variables after each line of code:

```
a) myVar ← 3
b) yourVar ← myVar + 2
c) myVar ← yourVar
```

Variables can be given any suitable value, but only one value can be stored in a variable at one time.

The code `myVar ← myVar + 1` takes the value in `myVar`, adds 1 to it and then overwrites the value in `myVar` with the new value. The order of the operations is important here. The addition takes place in another area of memory and the new value for `myVar` is then stored in the variable, overwriting the existing value.

✎ Exercise 2.3

The variables myVar and yourVar are used in this pseudo-code. Write down the values of both variables after each line of code:

```
a) myVar ← 65
b) yourVar ← "Clive"
c) myVar ← myVar + 5
d) yourVar ← myVar - 1
e) myVar ← "Anna"
```

Assigning in Python

When programming using Python, values are assigned to variables using the operator **=**. This is called the **assignment operator** and is only ever used in Python when assigning values to variables.

💻 Programming exercise 2.1

Use Python to write the following program. Run it and check the output.

```
# program to show assignment
myVar = 65
yourVar = "Clive"
print(myVar)
print(yourVar)
```

The purpose of the first line is to write a **comment**. Comments are used to leave messages for other programmers. Computers ignore them. It is good practice to include comments in all of your programs.

💻 Programming exercise 2.2

What values will be stored in myVar and yourVar at the end of this program?

```
# program to show changing values after assignment
myVar = 27
yourVar = 5
print(myVar)
print(yourVar)
print() # prints a blank line
yourVar = yourVar + 2
myVar = myVar + yourVar
print(myVar)
print(yourVar)
```

Don't forget that you need to know pseudo-code commands in the exam.

✎ Exercise 2.4

When using Python, print() is used to display values on screen. What is the command used in pseudo-code to do the same thing?

✎ Exercise 2.5

Explain why it is important to be able to understand psuedo-code commands

---

Programming exercise 2.3

Take the following pseudo-code algorithm and turn it into a suitable Python program:

```
myName ← "Elsa"
myAge ← 23
myAge ← myAge + 1
OUTPUT myName
OUTPUT myAge
```

3 Using user input

So far all the values for variables have been assigned as part of the program. This isn't all that helpful. It is often useful to allow users to enter their own values using the keyboard.

The pseudo-code command to do this is USERINPUT.

```
myVar ← USERINPUT
```

This assigns whatever value the user inputs to the variable myVar.

Exercise 3.1

Consider the following pseudo-code.

```
OUTPUT "Enter your user name"
userName ← USERINPUT
```

a) Suggest a suitable value to be inputted by the user
b) Write a line of pseudo-code to display the value which has been inputted on the screen at the end of the program

USERINPUT is almost always used to assign a value to a variable.

In Python, the command input() is used to do the same thing

💻 Programming exercise 3.1

Try this program.

```
# program to show user input
print("Enter your user name")
userName = input()
print(userName)
```

The first two lines of this program are usually combined into one:

💻 Programming exercise 3.2

```
userName = input("Enter your user name: ")
print(userName)
```

Note that when using pseudo-code the two lines of the input statement need to be written separately – you can't combine OUTPUT and USERINPUT on one line.

Summary

Variables are named areas of memory in which values can be stored. Once a variable has been **initialised** and **assigned** a value it can be changed as many times as needed.

Variables are given names called **identifiers**. Identifiers need to keep to basic rules. Programmers often use **camel case** when choosing variable identifiers to make things easier for other people to understand.

In pseudo-code the symbol ← is used to assign a value to a variable. In Python the symbol = is used to do the same thing.

Users can input values which can be assigned to variables.

Key terms:

- variable
- variable identifier
- initialisation
- assignment

Solutions to exercises:

Exercise 1.1:

a) either the name of the user or their username on a computer system would be sensible values to assign to a variable with this name

b) price or itemPrice would be sensible names to use as both are clearly related to the purpose of the variable. The identifier item doesn't make clear that the price is being referenced, value is vague and variable is far too vague

c) totalPrice would be a sensible identifier to use

Exercise 1.2:

a)

(i) lastName
(ii) addressTwo or addressLineTwo
(iii) totalPrice
(iv) phoneNumber or phoneNo or phone would all be reasonable names

b) It starts with a capital letter. Programmers don't usually use a capital at the start of a variable identifier. The reason for this is because other ideas used in computer programming use capitalised first letters and an experienced programmer would assume that you were using something more complex than a simple variable.

Exercise 1.3:

price is the only suitable identifier. input and while are both commands used in Python, while 4name starts with a number which is not allowed when creating variable identifiers in Python.

Exercise 2.1:

```
myAge ← 27
```

Exercise 2.2:

a) myVar is 3; yourVar does not yet exist and so has no value

b) myVar is 3; yourVar is 5

c) myVar is 5; yourVar is 5 – the value of yourVar has been assigned to myVar so they both have the same value, although stored in different areas of memory.

Exercise 2.3:

a) myVar 65; yourVar does not exist

b) myVar 65; yourVar "Clive"

c) myVar 70; yourVar "Clive"

d) myVar 70; yourVar 69

e) myVar "Anna"; yourVar 69

Exercise 2.4:

OUTPUT is the pseudo-code version of print()

Exercise 2.5:

Because exam questions will be written using pseudo-code. Note that you can write your answers in any form you want (including in Python) so long as they are clear. But you need to understand what the pseudo-code means to begin with.

Exercise 3.1:

a) you might have suggested your school user name or one you use on another system

b) OUTPUT userName