



Chapter 3

Computer Language

Look at Fig 3.1. Can you identify the difference in Sandeep's routine during the holidays?

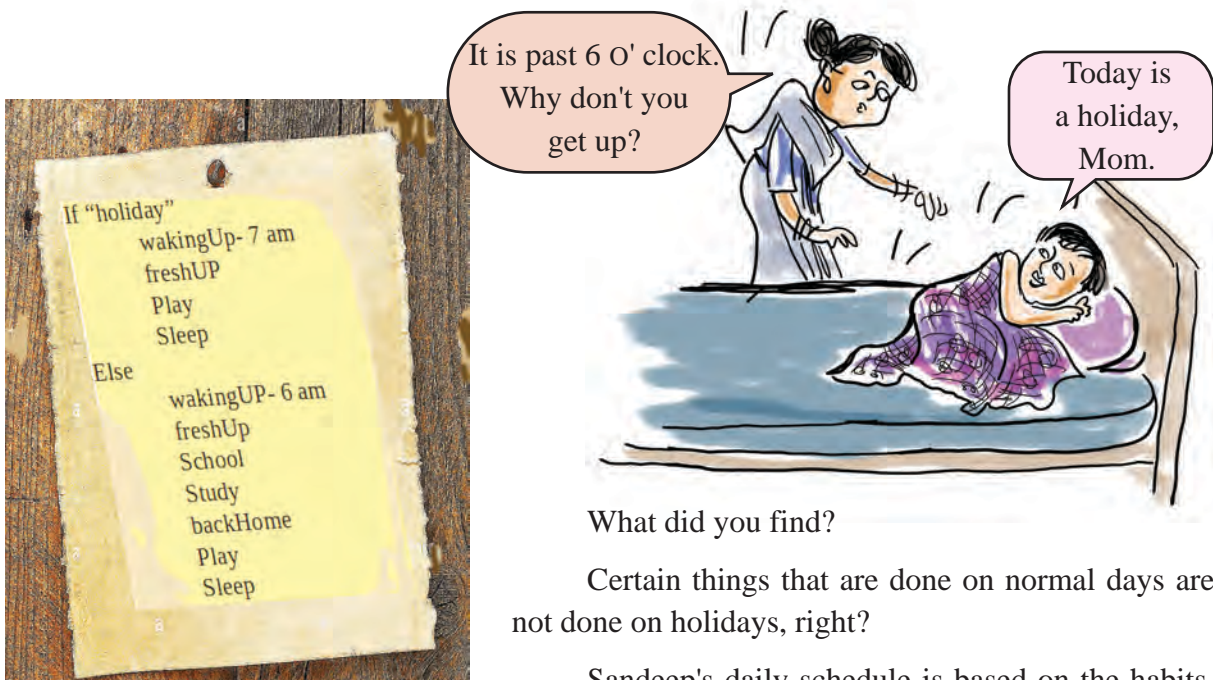


Fig 3.1 Schedule of routine

What did you find?

Certain things that are done on normal days are not done on holidays, right?

Sandeep's daily schedule is based on the habits, suggestions and knowledge he gained from various sources.

Aren't we the same?

The knowledge and habits gained from experiences and studies, along with the advice or instructions we receive from others, lead us.

Don't you know that computers also work based on the instructions given to them and the knowledge they gain from those instructions?

How do computers understand the instructions we give them?

A computer is an electronic device. Like any electronic device, computers only know ON and OFF states.

Instructions are given to computers through **binary language** codes that only use the symbols 0 and 1 to represent **ON** and **OFF** states.

But we don't know binary language. Then how can we give instructions to computer in binary language?

Imagine you have a friend who only speaks Bengali, and you don't know the language. How would you communicate with them?

A mutual friend who understands both Bengali and Malayalam could help you communicate effectively.

In this way, our computers have programs that can translate instructions given in a language into binary language. These are known as high-level languages and translator programs.

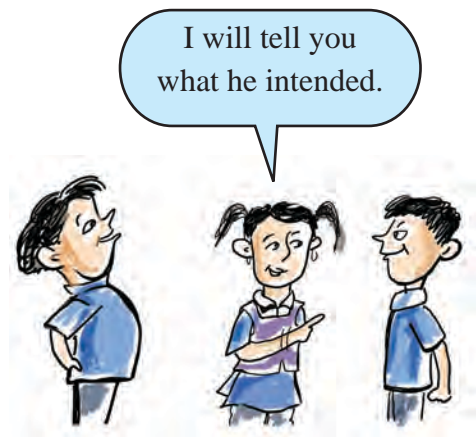
Translator programs are mainly of two types - **Compilers and Interpreters**.

Computers with the help of translator programs understand and act according to the high-level language instructions that we give.

Binary Language

A language that uses only **ON** and **OFF** is called a binary language.

These states are denoted as *High*, *Low* or *1*, *0* or *True* and *False*.



Compilers and Interpreters

Compilers are translators that completely convert instructions into binary language and compile them into a separate file. In contrast, interpreters process and execute each line of code one at a time during execution.

Binary files (executable files) created by compilers do not require the original high-level language code (source code) to run later. However, when using interpreters, the source code must be present on the computer to execute the program.

High-level programming languages such as C, C++, and Java use compilers to convert and run programs. Python, on the other hand, is an interpreted programming language.

Let's get familiar with such a high-level language in this lesson.

Have you noticed the daily routine of Sandeep given at the beginning of the lesson? Sandeep is working on the instructions received one by one for each day.

Similarly, computers can be given precise instructions to solve a problem and thus can be used for problem determining and solving. In this way, it is said that programming is giving the necessary instructions to computers to do a specific work.(Task).

For example, suppose the computer needs to be given instructions to prepare a program to calculate the age of a person.

What are the information needed to determine someone's age?

- You should know in which year the person is born.
-

What is the age of a person whose year of birth is 2011?

If the current year is 2025, the person will be 14 years old, right? How did you find it?

Just subtract the year of birth from the current year.

That is,

Age = current year - year of birth

If the above mentioned things are written step by step,

- Current year (current_year)=2025
- Birth year (birth_year)=2011
- Age = Current year - Birth year
(age=current_year – birth_year)
- Display the age on the computer.

The process of writing the steps for solving a problem in a specific order is called an **algorithm**.

Let's prepare a computer program to calculate the age of a person according to the above given steps.

Which computer language should we choose to prepare this program?

Many languages are available today for developing computer programs.

Which are the computer languages you have heard of?

-
-
-
-
-
-

Java, C, C++, Python, Ruby, PHP and many other programming languages are currently available for programming. Here we use the language **Python** for making programs.

To give instructions to the computer we need to know programming language.



PYTHON

Python is a suitable programming language for everyone—from young beginners exploring programming to experts working on advanced concepts like robotics and data science. Its simple syntax makes it easy to learn and use.



Fig 3.2 Guido van Rossum



Fig 3.3 Python Logo

The Python language was developed in 1990 by Guido van Rossum (Fig 3.2), a computer engineer from the Netherlands. Python is an open-source programming language.

Lets Make a Python Program

Let's see how to make the program for calculating age that we mentioned earlier using Python.

For this we have to use the Text Editor application on our computer. A Python interpreter is required to translate the prepared program into binary language.

Look at the program given below. The previously written algorithmic tasks are converted to Python language here. Open the application called **Text Editor** on the computer and type the program, without making mistakes. Then save it in your folder.

```
birth_year=2011
current_year=2025
age = current_year - birth_year
print("Your age is : ", age)
```


Program files written in Python should have the extension .py added to their file names. For example, the age finder program can be saved as Find_age.py.

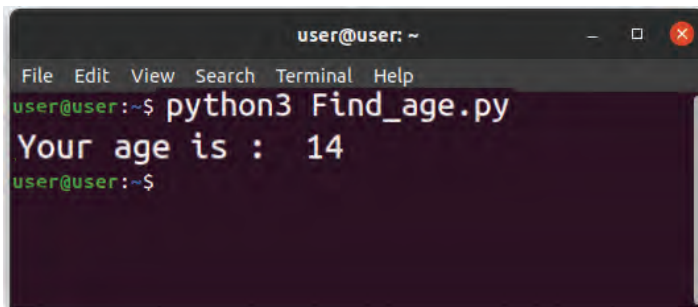
Isn't the file saved? Now try to run the file Find_age.py using the Python3.

To Run Python File

- Open the folder where the file is saved.
- Right click on the empty space in the folder. Click on **Open in Terminal**.
- Then, type the below given command in the terminal and press **Enter**.

```
python3 Find_age.py
```

See that when the program is run, the age is printed on the terminal. (Fig 3.4)



```

user@user: ~
File Edit View Search Terminal Help
user@user:~$ python3 Find_age.py
Your age is : 14
user@user:~$

```

Fig 3.4 Terminal Window

Check and find out how the line print("Your age is :", age) is run in this program.

After printing the text 'Your age is : ' the calculated value is printed in place of age.

A similar program to find the area of a square of fixed length and width is given below.

```
length=35
```

```
breadth=65
```

```
.....
```

```
print("Area of Rectangle is: ", area)
```

In Scratch,
instructions are given
for sprites as well.



Variables

Just like in mathematics, variables can also be used in computer programs. They are used to temporarily store data required for the program to run.

When using variables in a program, their names should start with a letter (alphabet) and must not include special characters, except underscore (_).

The term **Datatype** indicates what type of data each variable contains. Examples of datatypes include integers, floats, strings, images, audios, dates, and times. However, in Python, you don't need to specify the datatype when creating variables.

What should be added to the missing line in this program?

Open a new document in a **Text Editor**, complete the program, **save** and **run** it.

When the program was run to calculate age, we got the output based on the year we entered in the program. Anyone running this program at any time will get the output "14 years."

What should be done to convert this to a program that displays the age of any person who runs the program ?

If it is possible to input the current year and birth year to the program while running the program, it can be used anytime to calculate anyone's age, right?

Let's see how this program can be modified by giving name, birth year and current year of any person as input while running the program.

Instead of the statement `birth_year=2011` in the above program, try the following statement.

```
birth_year=input("Enter your year of birth:")
```

After changing the program, run it using Terminal.

What did you find?

.....

.....

In this way, let's add the following lines at the beginning of the program.

```
name=input( " What is your name?:")
```

```
current_year=input("Enter current year:")
```

When we run this program it asks for your birth year, name and current year and stores the birth year,

name and current year typed by keyboard in the variables *birth_year*, name and *current_year* respectively.

Save and **run** this program.

Did you get the output when made changes in this program?

This problem arises because mathematical operations are not possible with the inputs we have given. Difference can be calculated only if the data obtained as input is converted into numbers.

While Using the Input Function

In Python, data obtained using the input function is in text format, also known as a string.

Information like a person's name or address is typically made up of alphabetical characters, while details such as year of birth, height, or weight are represented as numbers.

Mathematical operations like addition, subtraction, multiplication, and division cannot be performed on text. Therefore, numbers in text format must be converted to numeric format for calculations.

The function `int()` can be used to convert data available in String form to integer form.

Let's rewrite the program as given below accordingly.

```
name = input("What is your name:")
current_year = input("Enter current year:")
birth_year = input("Enter your birth year:")
age=int(current_year) - int(birth_year)
print(name, " Your age is : ", age)
```

Note that the **int()** function is used in the line where the calculation is taking place.

Python Operators

Operators that can be applied to different data and variables in Python.

1. Arithmetic Operators:- For doing mathematical operations

Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%

2. Assignment Operator:- to give a value to a variable = is used.

Eg; Mark=50

3. Comparison Operators:- To compare values:

Equal to	==
Greater than	>
Less than	<
Greater than or equal	>=
Less than or equal	<=
Not equal	!=

4. Logical Operators:- For Combining multiple statements :

and, or, not

Python IDE, i.e., IDLE

We made the program in **Text Editor**. However, when preparing large computer programs, special software known as **IDE (Integrated Development Environment)** is used to help **run** the program easily and fix errors. The window of IDLE3, an IDE for writing programs in Python, is given in Fig 3.5.

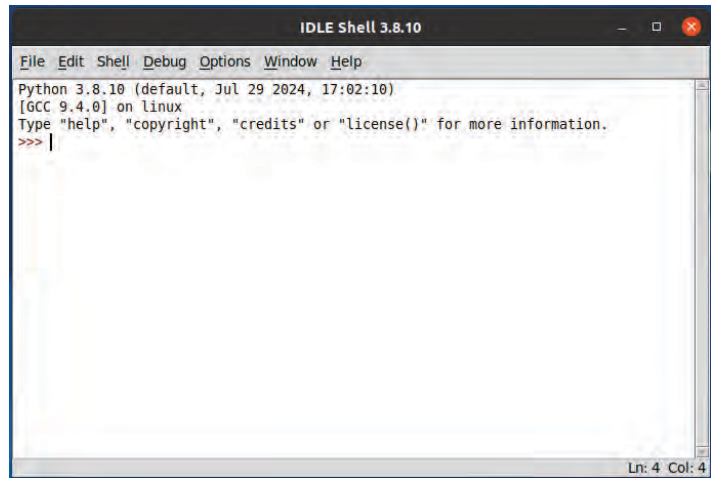


Fig 3.5 IDLE 3 Window

Open **IDLE3** on your computer and check what features are available in it.

Don't you see features that help you create a new file, **save** it, and **run** the program?

Let's Run the Program in IDLE3

The program that we prepared earlier was **run** in the terminal. Prepare and **run** the program in IDLE3 as shown below.

To run the Program in IDLE3

- Open IDLE3 and open a new editor using the option **File → New File**.
- Type the program without making mistakes. (Fig 3.6)
- Give file name and save the file using the option **File → Save**.

- Then **run** the program by clicking **Run Module** from the **Run** menu.

```
name=input("Enter your name:")
birth_year=input("Enter your birth year:")
current_year=input("Enter current year:")
age= int(current_year)-int(birth_year)
print(name, " Your age is : " , age)
```

Fig 3.6 Program made in IDLE3

```
File Edit Shell Debug Options Window Help
Python 3.8.10 (default, Jul 29 2024, 17:00:00) on linux
Type "help", "copyright", "credits" or >>>
>>>
===== RESTART: /home/
Enter your name:Sandeep
Enter your birth year:2011
Enter current year:2025
Sandeep Your age is : 14
>>> |
```

Fig 3.7 Output Window of the Program Made in IDLE3

While running the program name, year of birth and current year are given as input. Didn't you get the output according to the given data?

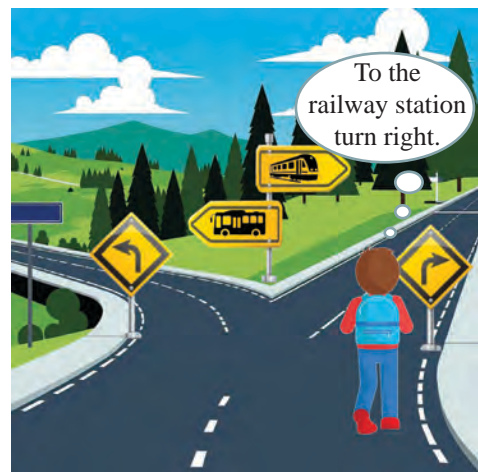
Conditional Statement

Didn't you see from the schedule given at the beginning of the lesson that Sandeep gets up and does other things on holidays unlike normal days?

That means, in certain days he works in one way and in other days he works in another way. Similarly in computer also, in some cases the flow of the program has to be varied depending on the conditions .

This is referred to as branching in programming methodology.

For example, during our school's sports fair, children compete in different categories based on



Syntax

Each programming language has its own terminologies and rules for writing programs. This is called **syntax**.

Pay special attention to the indentation of the line after **if <condition>**: which is executed when the condition is met, and line after **else**: statement which is executed when the condition is not met.

Python statements should be written using lowercase letters of the English alphabet. It is part of the syntax of Python language.

their age. Those aged 14 years and above are placed in the junior category, while those under 14 fall into the sub-junior category. Let's see how we can modify the program to display the category based on the age calculated in the program.

The **if...else** statement (condition) is used to check whether a specific condition is satisfied or not. If the condition is met, the program executes one set of instructions; otherwise, it follows a different path.

The way to type **if...else** statement is given below.

```
if <condition> :
```

```
    statements for condition true
```

```
else:
```

```
    statements for condition false
```

In our program, if the age is 14 years or older, the program must print the category as Junior, otherwise print the category as Sub-Junior. Then, It can be written using conditional statement as given below.

```
if age >= 14:
```

```
    print("Your are in Junior Category")
```

```
else:
```

```
    print("You are in Sub-Junior Category")
```

Let's add these lines and run the previously prepared program.

```
name=input("Enter your name:")
```

```
birth_year=input("Enter your birth year:")
```

```
current_year=input("Enter current year:")
```

```
age= int(current_year)-int(birth_year)
```

```
print(name, " Your age is : ", age)
```

```
if age >= 14:
```

```
    print("Your are in Junior Category")
```

else:

print("You are in Sub-Junior Category")

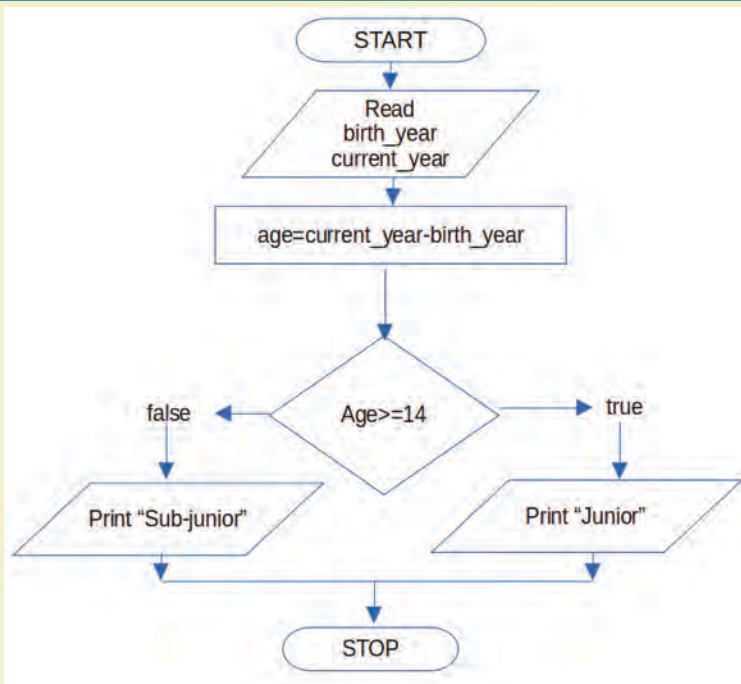
```
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.3.0]
Type "help", "copyright", "credits" or "license()"
>>>
===== RESTART: /home/
Enter your name:Anu
Enter your birth year:2013
Enter current year:2025
Anu Your Age is : 12
You are in Sub-Junior Category
>>>
```

Fig 3.8 Output of the Program Using *if...else*



Flow Chart

Look at the pictorial representation (Flowchart) of the program that we made. Such a diagrammatic representation is helpful in analysing and solving large and complex problems easily.



Here, we have checked only one condition. In some cases, if the first condition is not met, it is to be checked if other conditions are satisfied.

For example, those above 16 years should be categorized as Senior, those above 14 as Junior, those above 12 as Sub-Junior, and those below 12 as Kiddies.

To check these multiple conditions, the **if...elif...else** format can be used, as shown below.

```
if age >= 16:
    print("You are in Senior Category")
elif age >= 14:
    print("You are in Junior Category")
elif age >= 12:
    print("You are in Sub-Junior Category")
else:
    print("You are in kiddies Category")
```

Modify and run the program accordingly.

Loop Statements

Now, our program can calculate a person's age by taking the name, year of birth and current year and print the age category.

Suppose more than one child from a class participates in the school sports meet. What can we do to change this program to find the age and category of all of them?

The number of lines should repeat, as many times as the number of children participate in the competition, right?

Loops statements can be used to give instructions in this case.

There are mainly two types of loops in Python.

1. while loop
2. for loop

while loop

The syntax of while loops is as follows.

```
while <condition is true> :
    statements to repeat
```



Suppose three children from our class are participating in the sports fair. Let's see how to do it using a **while loop**.

We can add a variable called *count* in the program. Initially its value can be set to zero.

```
count = 0
```

For each repetition, the value of the variable *count* is to be incremented by 1.

For this we can use the code

```
count=count + 1
```

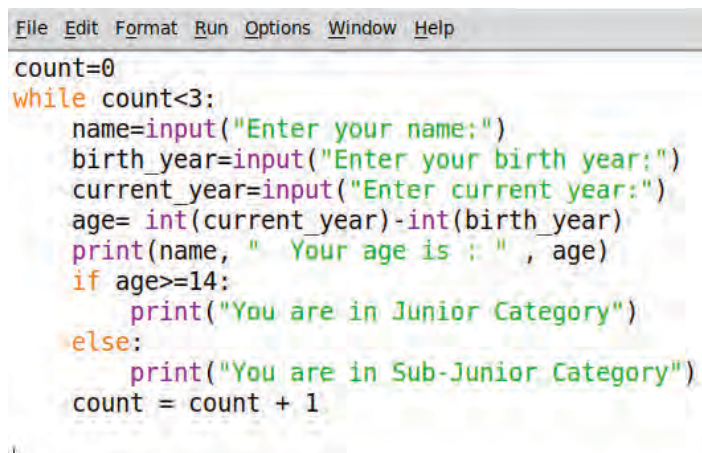
A total of three children are participating. Only when the count is less than three it should repeat. If it is 3 or greater than 3 the program should end.

Then the condition for checking statements in while loop is *count < 3*.

That is,

```
while count < 3 :
```

Now add the required codes and rewrite the program as shown in figure 3.9.



```
File Edit Format Run Options Window Help
count=0
while count<3:
    name=input("Enter your name:")
    birth_year=input("Enter your birth year:")
    current_year=input("Enter current year:")
    age= int(current_year)-int(birth_year)
    print(name, " Your age is : " , age)
    if age>=14:
        print("You are in Junior Category")
    else:
        print("You are in Sub-Junior Category")
    count = count + 1
```

Fig 3.9 Python Program Using *while* Statement.

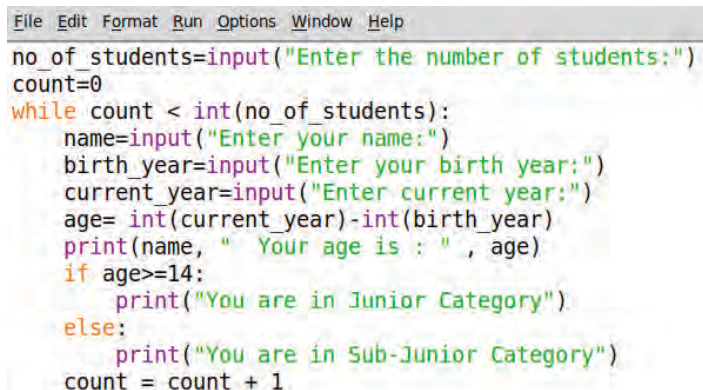
Did you note that statements to repeat are written one indentation away from the while block?

Now **run** the program.

The program ends when three children's data is entered, right? Why is it so?

- When the program starts, the value of the variable *count* is zero.
- Each time the code in the *while* block is executed, the value of *count* is incremented by one.
- When repeated three times, the value of *count* becomes 3. Then the condition *count* < 3 is not met and the program ends.

Consider rewriting the program to accept number of children as input when this program runs (Fig 3.10). The variable *no_of_students* is used here to take the number of students as input.



```
File Edit Format Run Options Window Help
no_of_students=input("Enter the number of students:")
count=0
while count < int(no_of_students):
    name=input("Enter your name:")
    birth_year=input("Enter your birth year:")
    current_year=input("Enter current year:")
    age= int(current_year)-int(birth_year)
    print(name, " Your age is : " , age)
    if age>=14:
        print("You are in Junior Category")
    else:
        print("You are in Sub-Junior Category")
    count = count + 1
```

Fig 3.10 Program that Accepts the Number of Students as Input

Difference between =, ==

= operator is used to assign a value to the variable

Eg: price=200 means to add the value 200 to the variable price.

The == operator is used to check whether the right and left sides are equal.

Eg: price==200 checks if price is 200.

for loop

Another technique for handling iteration statements is for loop.

Consider a Python program prepared using a **while** loop to print the numbers from 1 to 100 on the screen.

```
count=1
while count <=100:
    print(count)
    count = count +1
```

Now write and **run** this program in IDLE3.

Now, see the same program written using a **for** loop.

```
for count in range(1,101):
    print(count)
```

Did you type and run the program?

What are the differences between while and for loops? See Table 3.1.

while	for
Initial value is given to the variable count. (<i>count=1</i>)	All these are indicated in one line.
Given the code to increment the value (<i>count=count+1</i>)	for is used along with range. <i>for count in range(1,101):</i>
Given Code to check condition. (<i>count<=100</i>)	

Table 3.1 The difference between using *while* and *for* loops.

The Statement *range*

- In Python, *range* is used to arrange a set of numbers systematically.
- If *range (10)* is given, the decimal numbers from 0 to 9 (0, 1,2,3,4,5,6,7,8,9) are available for further computation.
- *range(1,10)* gives the 9 numbers from 1 to 9 (1,2.,3,4.5,6,7,8,9).
- *range(1,20,2)* gives the odd numbers from 1 to less than 20 (1, 3, 5, 7,9, 11,13, 15, 17, 19).
- That means the sequence of numbers will increase by two.

Now we have come across some programs prepared using Python language.

As we have just done, is it possible to get text output only using Python?

For example, if you want to draw a picture on a computer, what software do you usually use?

- Gimp
- Inkscape
-

Can you run a program that draws a picture? let's examine.

Python Graphics

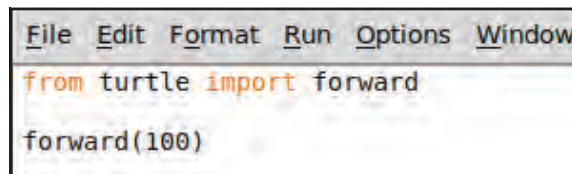
Not only text, but Python can also draw images and geometric shapes, by providing graphical output as well. Python has library modules to meet various needs. By using the turtle graphics module, you can create programs that generate graphical output in Python.

To draw a line in turtle, we use the command forward.

If we are using this single command ,

from turtle import forward can be typed at the beginning.

Open IDLE3 and run the python program given below (Fig. 3.11).



```
File Edit Format Run Options Window
from turtle import forward
forward(100)
```

Fig 3.11 Python program for drawing a line.

Python Modules

Modules are Python files that contain functions, variables, and classes for specific purposes. Modules can be included in the program as needed using the keyword import.

Some python modules

os - For functions related to the operating system

math - For mathematical operations

Eg: *sqrt, sin, cos* etc

turtle - graphics, drawing

datetime - those things related to time and date

A 100 pixel long line appears in the **graphics window** (Fig 3.12).

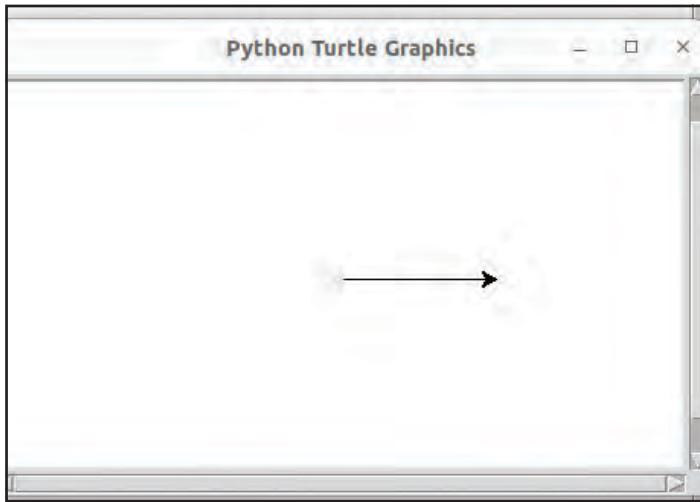


Fig 3.12 Python Graphics Window.

Add the statement `from turtle import *` at the beginning of the program to add all the functions and classes available in the turtle module to our program.

You have seen that the forward instruction can be used to draw a line.

The command `right(90)` can be used to rotate 90 degrees to the right.

If so, what instructions should be given to draw the shape of a playground with 100 units long and 100 units wide? (Fig 3.13).

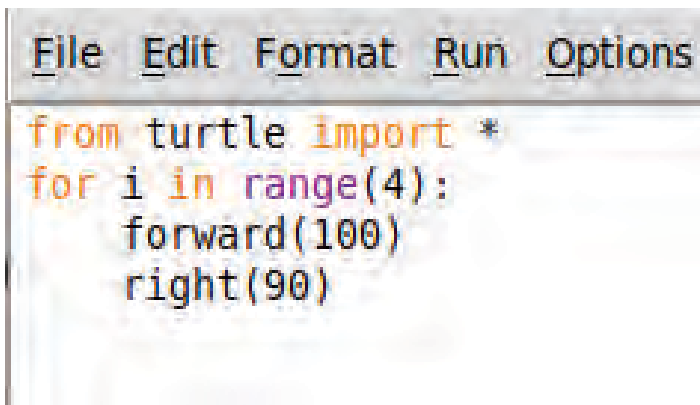


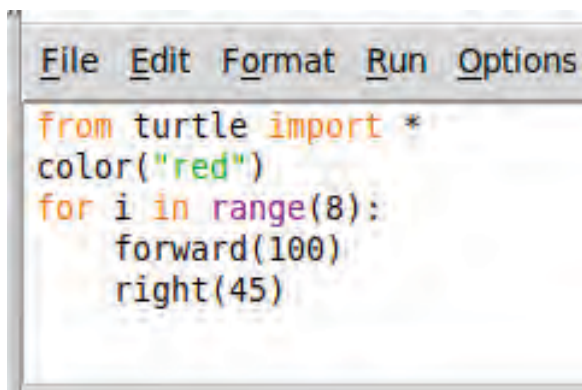
Fig 3.13 Program to Draw Playground with 100 Unit Length and 100 Unit Width.



In this program, the instructions to move forward 100 units and turn right 90 degrees are executed four times to produce the shape of a rectangular playing ground.

What colour are the shapes we have now?

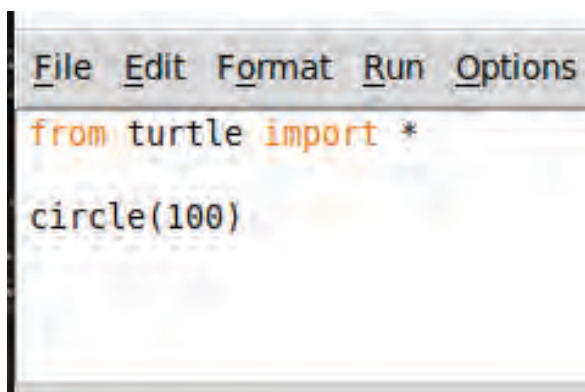
Open **IDLE3** and try typing and running the programs shown in Figures 3.14 and 3.15 one by one.



```
File Edit Format Run Options
from turtle import *
color("red")
for i in range(4):
    forward(100)
    right(90)
```

Fig 3.14 Program 1 to draw shape using turtle graphics

We can draw patterns easily using Python Graphics.

```
File Edit Format Run Options
from turtle import *
circle(100)
```

Fig 3.15 Program 2 to Draw Shape Using turtle graphics

See the outputs of the programs. Did you notice the use of *forward*, *left*, *right*, *circle*, *range* commands ?

Now, use these programs to draw different shapes and observe the output and complete Table 3.2.

Make changes in the colour of the line, length, number of sides, angle and radius of the circle .

Python Code	Use
<code>color("red")</code>	
<code>circle(100)</code>	To draw a circle of radius 100 unit.
<code>left(45)</code>	
<code>right(45)</code>	
<code>range(8)</code>	

Table 3.2 Python Graphic Codes and Uses

Lists in Python

Using the option **list** in Python, it is possible to store different data in a single string.

See how the names of the various colours are included in the variable colors.

```
colors = ["red", "yellow", "blue", "green", "orange", "violet"]
```

The data in a list can easily be used in a program that uses the for loop command. Try running the program given below.

```
for clr in colors:
```

```
    print(clr)
```

A Python program to create circles of different colours and radius is given in Fig 3.16. Type this program and run it and observe the output.



```

File Edit Format Run Options Window Help
from turtle import *

colors=["red", "yellow", "blue", "green", "orange", "violet"]

radius=20

for c in colors:
    color(c)
    circle(radius)
    radius=radius + 10

```

Fig 3.16 Python Program for creating Circles with different colours and radius



Let's Assess

- ♦ a=100
a=a+25
print(a)
What will be the output of this program?
- ♦ Which is the python program to **print** even numbers from 20 to 40?
 - a) for i in range(20,42,2):
 print(i)
 - b) for i in range(20,2,42):
 print(i)
 - c) for i in range(20,42):
 print(i)
 - d) for i in range(1,2,20):
 print(i)



Extended Activities

1. Create and print various lists in Python.
 - Names of friends
 - Names of flowers
 - Names of planets in Solar System

2. Write a Python program to find the area of a field if its length is 120 m and width is 50 m.
3. Write a Python program that takes three different numbers as input and print the largest number among them.
4. Take the score obtained by a student in a subject (out of 100 marks) as input and write a Python program to calculate the grade based on the following conditions.

Score \geq 90 Grade= A+

Score \geq 80 Grade=A

Score \geq 70 Grade=B+

Score \geq 60 Grade=B

Score \geq 50 Grade=C+

Otherwise print "Not Eligible"

5. Draw a rectangle with a length of 200 units and a width of 100 units, using blue, green, red, and yellow colours for the sides.

