Algorithmic Thinking with Python

SEMESTER S1(Common to All Branches)

Huda Noor Dean

College of Engineering Trikaripur

Outline

Introduction

A **program** is a set of instructions given to the computer to execute operations in succession that leads to solve specific problem. In general to solve any problem in computer we must follow these steps:

1. Analyze the problem.
2. Write an Algorithm.
3. Draw flowchart.
4. Convert the flowchart to program.
5. Run the program and test the solution.

Outline

Algorithms

- An algorithm describes a systematic way of solving a problem. It is a step-bystep procedure that produces an **output** when given the **necessary inputs**.
- An algorithm uses pure English phrases or sentences to describe the solution to a problem.
- An algorithm is a set of precise and unambiguous instructions that can be followed to achieve a specific goal or solve a particular problem.

Outline

Pseudocode

- A Pseudocode is a high-level representation of an algorithm that uses a mixture of natural language and programming language-like syntax.
- It is more structured and uses mathematical expressions along with English phrases.
- We can use programming constructs in a pseudocode but not in an algorithm.
- Pseudocode is not a true program and is independent of any programming language.
- It is not executable rather helps you understand the flow of an algorithm.

Algorithm and Pseudocode to add 2 numbers

ALGORITHM

1 Start
2 Read values of a, b.
3 Find the sum of a and b.
4 Store the sum in c.
5 Display the value of c.
6 Stop.

PSEUDOCODE

```
1 Start
2 Read a, b
3 c = a + b
4 Print(c)
5 Stop
```

Why pseudocodes?

1. **Ease of understanding:** Since the pseudocode is programming language independent, novice developers can also understand it very easily.
2. **Focus on logic:** A pseudocode allows to focus on the algorithm's logic without bothering about the syntax of a specific programming language.
3. **More legible:** Combining programming constructs with English phrases makes pseudocode more legible and conveys the logic precisely.

Why pseudocodes?

4. **Consistent:** As the constructs used in pseudocode are standardized, helps in sharing ideas among developers from different domains.
5. **Easy translation to a program:** Using programming constructs makes mapping pseudocode to a program easy.
6. **Identification of flaws:** A pseudocode helps identify flaws in the solution logic before implementation.

In general, we can divide pseudocode constructs as :

1. Sequence structures
2. Branched structures or decision statements
3. Repeation of Looping structures.

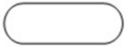
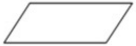
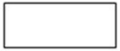
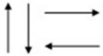
Outline

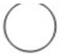
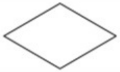
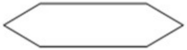
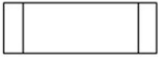
Flowchart

- A flowchart is a diagrammatic representation of an algorithm that depicts how control flows in it.
- Flowcharts are composed of various blocks interconnected by flow-lines.
- Each block in a flowchart represents some stage of processing in the algorithm.
- Different types of blocks are defined to represent the various programming constructs of the algorithm.

Flowchart

The following table shows these shapes and their operations.

Shape	Operation
	Start or End
	Input / Output data • Read or Input • Print
	Processing / Storing • Add(+) , Sub(-) , Mul(*) , Div(/) , Power(^), ... Store a value (Put)
	Flow Lines

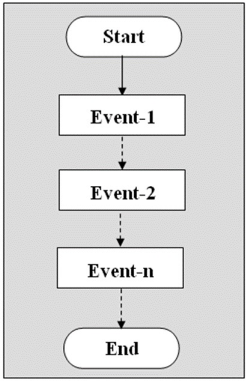
Shape	Operation
	Connection
	Decision • If statement • Question (?)
	Looping / Counters
	Subroutine

In general, we can divide flowcharts to a four :

1. Simple sequence charts
2. Branched charts.
3. Single loop charts.
4. Multi-loop charts.

Simple sequencing

The arrangement of event types in a straight sequence from the beginning of the program to the end (Event-1 to Event-n in figure below) is Simple sequence charts. This type of charts does not have any branches or loops.



Flow lines

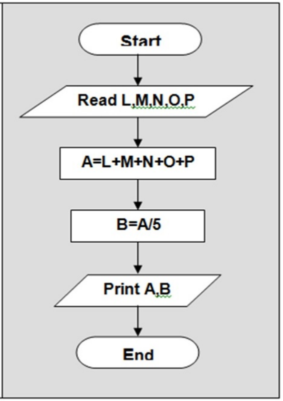
Flow lines indicate the order in which the algorithm steps are executed. The flow lines entering a block denote data (or control) flow into the block and the flow lines emerging from a block denote data (or control) flow out from the block.

Most blocks have only single incoming and outgoing flow lines. The exception is for blocks representing selection and loop constructs. Such blocks have multiple exits, one for each possible outcome of the condition being tested and each such outcome is called a **branch**.

Example 1: Find sum and average of 5 numbers

Solution:

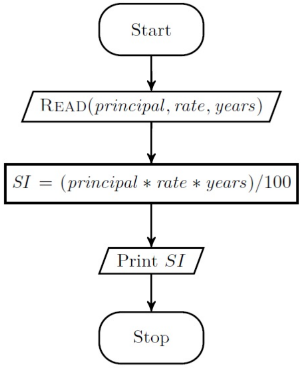
1. Start
2. Read L, M, N, O, P
3. Find Sum (A) , A=L+M+N+O+P
4. Find Average (B) , B= A / 5
5. Print A, B
6. End



Example 2:To find simple interest

SimpleInterest

```
1 Start
2 Read(principal , rate, years)
3 SI = (principal * rate * years) / 100
4 Print(SI )
5 Stop.
```


Start
READ(principal, rate, years)
SI = (principal * rate * years)/100
Print SI
Stop

Tutorials:

1.Write Algorithm and flowchart to find Area and Circumference of a circle.

2. Write an algorithm and flowchart to find the Area of a Triangle given base and height.

3. Create an algorithm and a flowchart to convert temperature from Fahrenheit to celsius.C=(F-32)*5/9

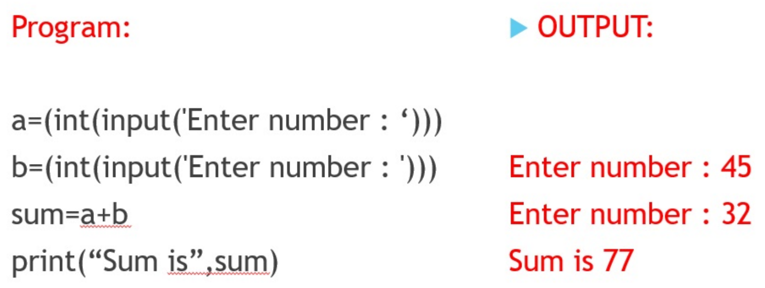
4. Write an algorithm and a flowchart to swap two variables with and without using a temporary variable.

Simple Sequence Programs

Write a program to find sum of two numbers

Here we think about the **input to be given** and then the **output to be obtained**, and then we find the **logic to be implemented** to get the result from the input.

WAP to find sum of 2 numbers



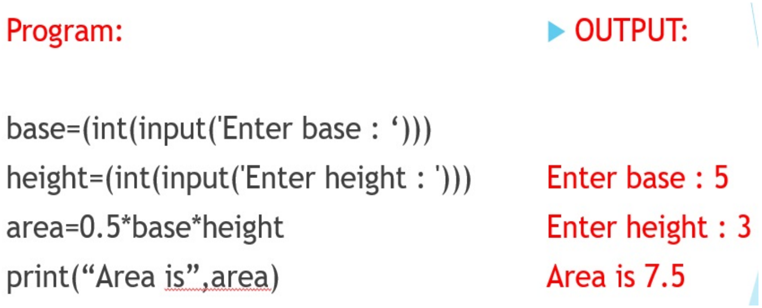
Program:

```
a=(int(input('Enter number : ‘)))
b=(int(input('Enter number : ')))
sum=a+b
print(“Sum is”,sum)
```

OUTPUT:

```
Enter number : 45
Enter number : 32
Sum is 77
```

WAP to find area of triangle given base and height



Program:

```
base=(int(input('Enter base : ‘)))
height=(int(input('Enter height : ')))
area=0.5*base*height
print(“Area is”,area)
```

OUTPUT:

```
Enter base : 5
Enter height : 3
Area is 7.5
```

Tutorials:

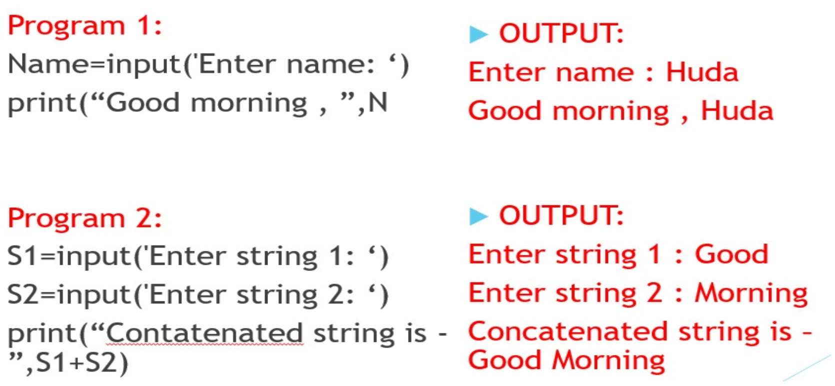
1. Write a program in C that takes minutes as input, and display the total number of hours and minutes.(Input minutes: 546, Output: 9 Hours, 6 Minutes)

2. WAP to find area and circumference of a circle.

3. Create an algorithm and a flowchart to convert temperature from Fahrenheit to celsius.C=(F-32)*5/9

4. WAP to swap two variables with and without using a temporary variable.

String programs



Program 1:

```
Name=input('Enter name: ‘)
print(“Good morning , ”,N
```

▶ OUTPUT:

```
Enter name : Huda
Good morning , Huda
```

Program 2:

```
S1=input('Enter string 1: ‘)
S2=input('Enter string 2: ‘)
print(“Contatenated string is -
”,S1+S2)
```

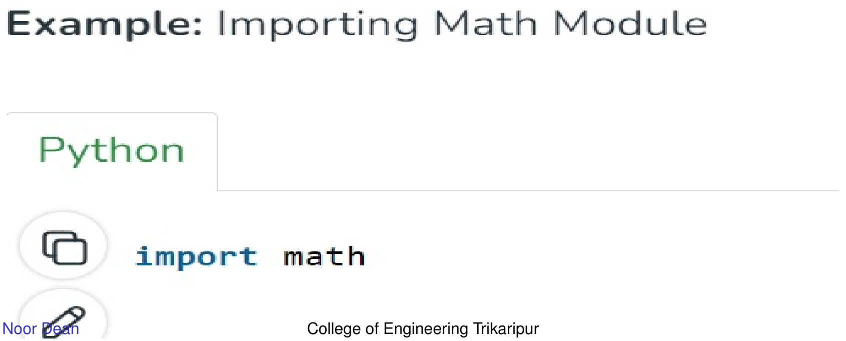
▶ OUTPUT:

```
Enter string 1 : Good
Enter string 2 : Morning
Concatenated string is -
Good Morning
```

Outline

Python math Module

Math Module is an in-built Python library made to simplify mathematical tasks in Python.
It consists of various mathematical constants and functions that can be used after importing the math module.



Example: Importing Math Module

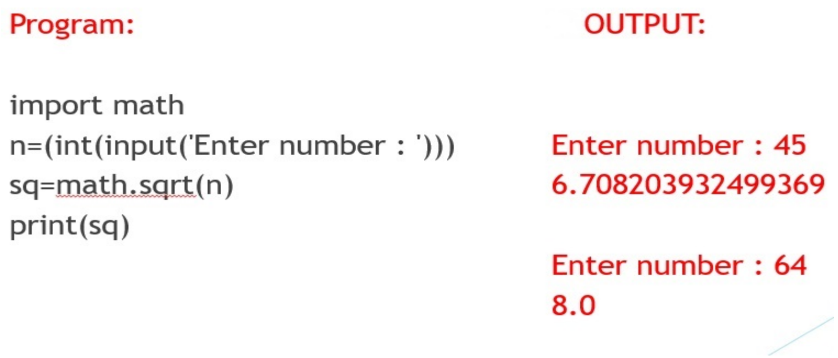
Python

```python
import math
```

Methods in Math Module

Function Name	Description
ceil(x)	Returns the smallest integral value greater than the number
fabs(x)	Returns the absolute value of the number
factorial(x)	Returns the factorial of the number
floor(x)	Returns the greatest integral value smaller than the number
gcd(x, y)	Compute the greatest common divisor of 2 numbers
exp(x)	Returns the value of e raised to the power x(e**x)
pow(x, y)	Compute value of x raised to the power y (x**y)
sqrt(x)	Returns the square root of the number

Find the square root of a number



Program:

```python
import math
n=(int(input('Enter number : ')))
sq=math.sqrt(n)
print(sq)
```

OUTPUT:

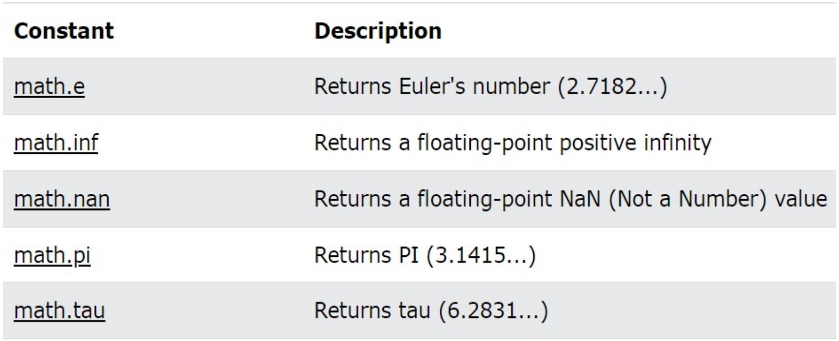
```
Enter number : 45
6.708203932499369

Enter number : 64
8.0
```

Tutorials

1. WAP to calculate hypotnuse of a right angled triangle.
2. WAP to calculate the area of a triangle(given 3 sides).
3. WAP to calculate the distance between two points.

Constants in Math Module

The Python math module provides various values of various constants like pi, and tau. We can easily write their values with these constants. The constants provided by the math module are :



Constant	**Description**
math.e	Returns Euler's number (2.7182...)
math.inf	Returns a floating-point positive infinity
math.nan	Returns a floating-point NaN (Not a Number) value
math.pi	Returns PI (3.1415...)
math.tau	Returns tau (6.2831...)

Find the area of the circle

The code utilizes the math module in Python, defines a radius and the mathematical constant pi, and calculates the area of a circle using the formula' A = pi * r * r'. It demonstrates the application of mathematical concepts and the usage of the math module for numerical calculations.

```
import math
r = int(input('Enter Radius'))
print('Area is ',math.pi * r * r)
```

Purpose of using Algorithms and Flowcharts

- To show the mathematical logic used to solve problems.
- To show how the data processing is done.
- Helps the programmer to write his program.
- Divides the bigger problem to smaller parts.
- It is a middle step between problem difficulty and its conversion to a suitable program.
- Easy to convert to any programming language.

Outline

Flowchart connectors

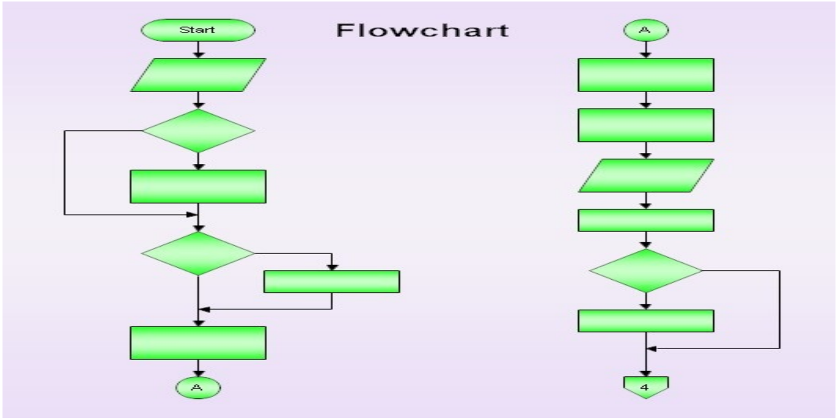
Sometimes, flowcharts becomes so expansive and complex that jotting all your ideas down on one page is impossible.
Flowchart connectors are the symbols employed for connecting two adjacent operations shifted to the next panel or page when running out of space during the flowchart creation. The connectors have two major types:

- **on-page flowchart connectors,** used to adjoin two or more operations on the same page, and
- **off-page flowchart connectors,** which connect the two or more operations drawn on different pages.

A flowchart connector works in pairs, requiring you to add each connector at the start and end of a certain operation.

On Page Flowchart Connectors

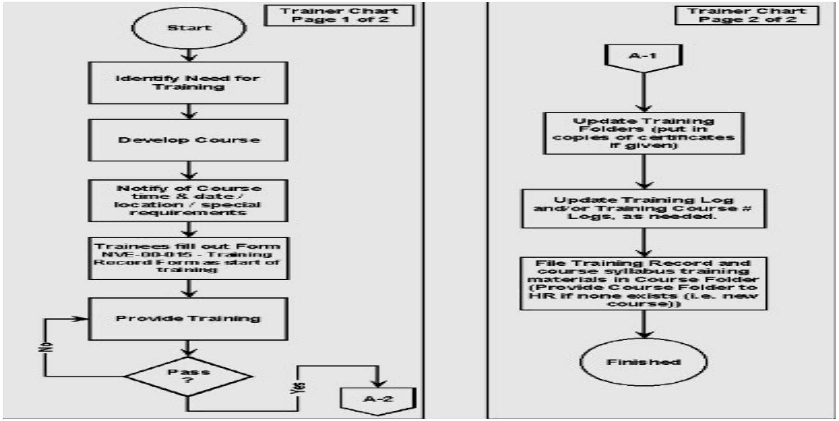
On-page flowchart connectors are those which are employed to link two ideas existing on the same page. For example, if you are making a linear flowchart, and the page ends, you can shift your remaining chart operations to the other side using the on-page flowchart connector symbol. The on-page flowchart connector symbol is a circle containing any specific letter which is inserted in both on-page connectors to demonstrate that the next one is the continuation of the previous.


Flowchart
Start
A
A
4

Off Page Flowchart Connectors

The off-page flowchart connector is utilized when you target to connect two different operations scattered over two variant pages. Whether linear or non-linear, if your flowchart diagram is super-vast, using the off-page flowchart connector is indispensable.

The off-page flowchart symbol is denoted by a semi-square, containing an arrow-like shape at the bottom.


Trainer Chart Page 1 of 2
Start
Identify Need for Training
Develop Course
Notify of Course time & date / location / special requirements
Trainees fill out Form NVE-00-015 - Training Record Form as start of training
Provide Training
Pass ?
No
Yes
A-2
Trainer Chart Page 2 of 2
A-1
Update Training Folders (put in copies of certificates if given)
Update Training Log and/or Training Course # Logs, as needed.
File Training Record and course syllabus training materials in Course Folder (Provide Course Folder to HR if none exists (i.e. new course))
Finished