PMDS508L - Python Programming
Module 1: Algorithmic Problem Solving

Dr. B.S.R.V. Prasad
Department of Mathematics
School of Advanced Sciences
Vellore Institute of Technology
Vellore

srvprasad.bh@gmail.com (Personal)
srvprasad.bh@vit.ac.in (Official)
+91-8220417476

Python Data Science Techniques

Algorithmic Thinking
Definition

- An **Algorithm** is sequence of discrete actions that when followed, will result in achieving some goal or solving some problem.
- In other words, an **Algorithm** is a sequence of clearly defined steps that describe a process to follow a finite set of unambiguous instructions with clear start and end points.

Algorithmic Thinking
Definition

- An **Algorithm** is sequence of discrete actions that when followed, will result in achieving some goal or solving some problem.
- In other words, an **Algorithm** is a sequence of clearly defined steps that describe a process to follow a finite set of unambiguous instructions with clear start and end points.
- Logic and algorithms are not same.
- Algorithms build on logic because, as part of their work, algorithms make logical decisions and stich those decisions together.

Algorithmic Thinking Examples

- Everyone is accustomed to following algorithms in daily life whether we are aware of it or not.
- For example:
  - Players are following an algorithms when they execute a play on the filed.
  - Drivers are using an algorithm when they follow a set of instructions for getting from one city to another.
  - Musicians follow a set of rhythmic instructions (algorithm) to produce music which is enjoyable by everyone.
  - Mathematicians use a step-by-step well defined and well executed steps to solve any mathematical problem such as finding the ratio of two numbers etc.

Algorithmic Thinking
Cookie Recipe

Chocolate Chip[1] Cookie Recipe

Ingredients

- 1 cup melted butter
- 2 cups brown sugar
- 2 eggs
- 3 cups flour
- 1 teaspoon baking powder
- 1 teaspoon baking soda
- 2 cups chocolate chips

Directions

1. Preheat the oven to 375 degrees F.
2. Line a cookie sheet with parchment paper
3. In a bowl, stir together the butter, brown sugar and eggs.
4. In a separate bowl, combine the flour, baking powder and baking soda. Gradually combine with the sugar mixture.
5. Add the chocolate chips
6. Fill the cookie sheet with one-spoonful drops of the cookie dough.
7. Bake dough for 9 minutes
8. Cool for five minutes before removing from cookie sheet.

Image courtesy of http://www.flickr.com/photos/amagill/34754258/

Algorithmic Thinking
Properties of Algorithms

- One of the most important requirement is that each of the individual actions referred to in the algorithm be meaningful.
  - i.e., programmers must know the precise meaning of each action they write.
- Computer programmers refer to the meaning of action as the *semantics* of an action.
- The phrase *semantics* refers to the meaning of the actions that occur in an algorithm.

- Another requirement is that actions of an algorithm must have only one possible interpretation.
  - i.e., there is no misunderstanding about the semantics of the action.
- We say that an action must be *unambiguous*; meaning that the action is not subject to conflicting interpretations.

Algorithmic Thinking
Properties of Algorithms

- An algorithm must also ensure that the order in which the actions occur be well defined.
- Finally, requirement is that the number of actions that are described in algorithm must be finite rather than infinite.
  - No goal can be reached if we are required to follow a never-ending sequence of actions.
  - In computer terminology this requirement can be expressed as every algorithm must halt in order to be useful.

Defining Algorithms

- Summarising the above we have the following:
    - Algorithm is a collection of individual steps
    - Each step must be precisely defined and has one and only one meaning i.e., unambiguous.
    - Algorithms are sequential i.e., the steps must be carried out in the order specified.

Components of Algorithm

An algorithm consists of

- Statements
- State
- Control Flow/Repetition
- Functions/Modularisation

Components of Algorithm Statements

- In programming, a **statement** is a single line of code that performs a specific action.
- It can be an assignment, a function call, a loop, or a conditional operation.
  - For example, an assignment statement like `x = 10` assigns the value `10` to the variable `x`.

Components of Algorithm
State

- **State** refers to the condition or status of a program or system at a particular moment.
- It represents the values stored in variables, memory, and other data structures.
  - For instance, if you have a counter variable `count`, its state could be the current value of `count`.

Components of Algorithm
State

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$ The state is now [Celsius = 33.5]

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $\mathit{Celsius} \leftarrow 33.5$ The state is now [Celsius = 33.5]
2. $\mathit{Fahrenheit} \leftarrow \mathit{Celsius} * 9$

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$ The state is now [Celsius = 33.5]
2. $Fahrenheit \leftarrow Celsius * 9$ The state is now [Celsius = 33.5 and Fahrenheit = 301.5]

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$ The state is now [Celsius = 33.5]
2. $Fahrenheit \leftarrow Celsius * 9$ The state is now [Celsius = 33.5 and Fahrenheit = 301.5]
3. $Fahrenheit \leftarrow Fahrenheit/5$

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$ The state is now [Celsius = 33.5]
2. $Fahrenheit \leftarrow Celsius * 9$ The state is now [Celsius = 33.5 and Fahrenheit = 301.5]
3. $Fahrenheit \leftarrow Fahrenheit/5$ The state is now [Celsius = 33.5 and Fahrenheit = 60.3]

Components of Algorithm
State

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$ The state is now [Celsius = 33.5]
2. $Fahrenheit \leftarrow Celsius * 9$ The state is now [Celsius = 33.5 and Fahrenheit = 301.5]
3. $Fahrenheit \leftarrow Fahrenheit/5$ The state is now [Celsius = 33.5 and Fahrenheit = 60.3]
4. $Fahrenheit \leftarrow Fahernheit + 32$

Components of Algorithm
State

Below we present an algorithm for converting temperature from Celsius to Fahrenheit.

$$\text{temperature in Fahrenheit} = \text{temperature in Celsius} \times \frac{9}{5} + 32$$

Convert from Celsius to Fahrenheit Algorithm

1. $Celsius \leftarrow 33.5$ The state is now [Celsius = 33.5]
2. $Fahrenheit \leftarrow Celsius * 9$ The state is now [Celsius = 33.5 and Fahrenheit = 301.5]
3. $Fahrenheit \leftarrow Fahrenheit/5$ The state is now [Celsius = 33.5 and Fahrenheit = 60.3]
4. $Fahrenheit \leftarrow Fahernheit + 32$
   The state is now [Celsius = 33.5 and Fahrenheit = 92.3]

Components of Algorithm
Control Flow

- **Control flow** is a computational term that refers to the specific order in which individual actions of a computer program are executed.
- Normally actions are performed sequentially in the order that they have been written.
- Control flow statements can change this ordering and control the specific ordering of the actions performed by a program.
- There are two types of control flows:
  - Selection statement
  - Repetition

Components of Algorithm
Control Flow - Selection

- A *selection statement* is a control flow that allows a program to make choices regarding whether certain actions should be performed.
- Selection statement again classified into three types:
  - *One-way selection* - allows a programmer to either perform an action or skip the action
  - *Two-way selection* - allows the computer to choose one of exactly two actions
  - *Multiway selection* - allows the computer to choose one of several alternatives

Components of Algorithm
Control Flow - Repetition

- We often required to repeat a sequence of actions in order to achieve some greater outcome.
- A *loop* is a control structure that repeatedly executes a sequence of actions.
- A *for loop* repeats a block of code a specified number of times.
- A *while loop* repeats a block of code as long as a condition is true.
- A *do-while loop* executes a block of code once and then repeats it while a condition holds.

Components of Algorithm
Control Flow - Repetition

All well-written loops must have three well-defined elements.

1. **Initialisation** — Every variable that occurs in the loop must hold the correct value prior to entering the loop.
2. **Condition** — The condition that determines when to repeat the loop must be precise.
3. **Progress** — The actions that are repeatedly executed must in some way make progress that allows the loop to terminate.

Components of Algorithm
Functions

- **Functions** are reusable blocks of code that perform specific tasks.
- They encapsulate logic, accept input (parameters), and produce output (return value).
  - For example, the factorial occurrence in $\binom{n}{r}$.

Pseudo-Code

- Algorithms are usually presented in the form called pseudo-code.
- A pseudo-code is an informal high-level textual description of the operating principle of a computer program or the algorithm.
- It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading.
- Good pseudo-code abstracts the algorithm, makes good use of mathematical notation and is easy to read.
- Bad pseudo-code gives too many details or is too implementation specific.

Pseudo Code

Algorithm: Maximum in a set of numbers

Data: A set $A = \{a_1, a_2, \ldots, a_n\}$ of integers
Result: An index i such that $a_i = \max\{a_1, a_2, \ldots, a_n\}$

```
begin
    index ← 1;
    for i ← 2 to n do
        if a_i > a_index then
            index ← i;
        end
    end
    output: index, a_index;
end
```

Pseudo Code

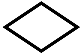
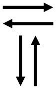
Algorithm: Maximum in a set of numbers

Data: A set $A = \{a_1, a_2, \ldots, a_n\}$ of integers
Result: An index i such that $a_i = \max\{a_1, a_2, \ldots, a_n\}$

```
begin
    index ← 1;
    i ← 1;
    while i ≤ n do
        if a_i > a_index then
            index ← i;
        end
        i ← i + 1;
    end
    output: index, a_index;
end
```

Flowcharts

- A graphical tool that *diagrammatically* depicts the steps and structure of an algorithm or program
- The most commonly used symbols of flow chart are listed below

Symbol	Name/Meaning	Symbol	Meaning
	Process – Any type of internal operation: data transformation, data movement, logic operation, etc.		Connector – connects sections of the flowchart, so that the diagram can maintain a smooth, linear flow
	Input/Output – input or output of data		Terminal – indicates start or end of the program or algorithm
	Decision – evaluates a condition or statement and branches depending on whether the evaluation is true or false		Flow lines – ***arrows*** that indicate the direction of the progression of the program

General Rules for Flowcharts

- All symbols of the flowcharts are connected by flow lines (note *arrows*, not by lines).
- Flow lines enter the top of the symbol and exit out the bottom, except for the Decision symbol, which can have flow lines existing from the bottom or the sides.
- Flowcharts are drawn so flow generally goes from top to bottom.
- The beginning and the end of the flowcharts is indicated using Terminal symbol.

Flowchart Example

Finding quotient when one number is divided by another