Preface

Dear friends,

This is an attempt to develop some lecture notes to assist you in “programming and problem solving using Python”. There has been a delay. No excuses. Blame on my laziness.

All the material collected and presented here is the contribution of the team offering the course – the ICUP team. Any omissions and errors are entirely mine.

Any constructive feedbacks to make this better are always welcome.

With warm regards,

N S Kumar
kumaradhara@gmail.com

Let us ask ourselves a few questions and answer them.

- **What is programming?**

Program is a sequence of instructions to solve a computational problem.

“Programming is a human activity” said Dijkstra – one of the greatest computer scientists ever. It is all about thinking. It is a creative activity. No two individuals will carry out an activity the same way.

There is a mythological story of Shiva Parvathi asking their children Ganesha and Skanda to go around the world and come back. Skanda goes on his motorbike (i. e. Peacock) round the world and Ganesha goes round his parents as he considers his parents to represent the world! Thus Ganesha provides a faster solution whereas my namesake fails (like me!).

Any creative, logical, thinking activity should be fun.

- **Why should we learn program?**

Let us watch the video : You Should Learn to Program: Christian Genco.

- We are considered literate today if and only if we know programming
- We can solve computational problems if and only if we know how to command the computer to do something for us. That requires programming skills.
- We want to sharpen our thinking skills.

- **What is a programming language?**

  It seems that young twins of my friend started developing their own way of communicating between them. It seems a doctor told my friend and his wife to separate the children lest they refuse to communicate with the parents!

  We require a language to communicate. Deaf-and-dumb use some sign language. Musicians have some interesting symbols to record the notes of a song – accepted and agreed upon by other fellow musicians.

  We require a language to communicate with our dumb computers. They do have their own language – similar to the languages small babies have! That is called the machine language.

  We find it very difficult to communicate to the computer in their language. We want to use a language which makes it easier for us to express our ideas. We want a language closer to the domain of application. These sorts of languages are called high level languages.

  A programming language provides the necessary constructs to instruct the computer to do something useful.

  A program is a sequence of instructions in a programming language.

- **How does a computer understand a program in a high level language?**

  It can not directly understand a program in a high level language as it only knows machine language.

  We require a mechanism to translate a program in a high level language to a program in a machine language. These are called translators.

  It is similar to what happens in the real world. A small child talks in a language which is understandable to its mother alone. She has to

translate the ‘child talk’ to her friends so that her friends can appreciate the intelligence of the child – does not really matter whether the child is a prodigy!

A translator can convert a program in a high level language completely at one go into a program in the machine language. Or it may convert in small chunks called statements. The former is called a compiler and the latter is called an interpreter. It is also possible to have translators which may convert bigger chunks or convert to some other intermediate form. These are hybrid models.

It is sufficient to realize that we require some translator to make the computer understand programs written in some high level language.

- **What is Python?**

  Python is a high level language developed by Guido Van Rossum. It is named after ‘Monty Python circus Show- a satirical show aired on BBC in the 60s and 70s. You may want to watch them on YouTube.

  Top 10 Monty Python Movie Moments – YouTube

  Python is a simple yet powerful language well received by both the academia and the Industry. We will revisit the characteristics of Python later in these lectures.

- **Why learn Python?**

  Life is all about selecting between the alternates. You must have debated before selecting Engineering over medicine, selecting college A over college B. I am sure you would have some criteria before choosing between the alternates – Your parents run a hospital, your uncle is a software specialist, your neighbours whom you do not like boast a lot about their child’s achievement or you tossed a coin to decide.

It is the same here too. We have umpteen number of languages to choose from. After lots of experiments with various languages, we have decided that you should start your journey in programming with Python. Recently Stanford also changed over to Python as the first language to be taught.

We prefer Python for a number of reasons. The four most important reasons are:

- simple to learn
- easy to master
- extensively used in the industry
- extensively used in research

Introduction to python programming:

There are two ways of making the computer follow the commands of Python.

1. interactive mode

    python

    >>> <here enter commands>

2. batch mode

    create a file of commands in python

    run them together

    python <filename>

In interactive mode, the computer will keep displaying the results immediately. If we give a formula (called expression), it will find the value and display it.

In batch mode, we should explicitly indicate that we want to display something using a special command(function).

check: 1_intro.py

A part of the file is shown for understanding.

```
3 * 4   # does not show anything on the screen
```

```
# use print for displaying
print(5 * 6)
# print : said to be a function
# does not give back anything - does not return anything
# it does display
```

Program structure:

We have used a function called print in the earlier program. The function concept is based on the concept of mathematics – example sine, cosine from trigonometry, distance formula given a pair of points. A function takes a number of arguments and gives a result. There are functions which will do something for us without returning any result.

print function takes a number of arguments, finds the value of these arguments and displays them on the computer screen. It does not give a value back.

We will now discuss the program structure.

Refer to 3_program_structure.py

- A program in python has number of statements. These statements are followed by the computer one after another in a sequence. We call this “executing a program” or “running a program”.

- Python distinguishes uppercase and lowercase characters – like ‘A’ and ‘a’. Most of the words we use in Python are in lower case.
  Print("nalku") # gives an error.

  Refer to 2_errors.py

- Language has grammar rules called syntax. If the syntax is violated, the Python translator gives an error.
  Example 1: A sequence of symbols(characters) is called a string. A string which does not change is referred to as a string constant or string literal. It has to be surrounded by quotes – single or double – should be same in

the beginning as well as the end. If quotes are missed at one end or not properly matched, then the translator gives an error.

```
print('this will be syntax error")
```

Example 2: All statements unless special construct is used should start from the first column. If the rule is not followed, the translator gives an error.

```
 print('another syntax error')
```

- If a program has no syntax error, then Python will execute the statements one by one. Sometimes we may get errors at run time. Then the program stops at that point and indicates the error. This is checked only when that point is reached during the program execution.
  Example : print(10/0)

- Each statement by default (normally) should be on a line by itself.
  - We can not normally have more than one statement on a single line
  - We can not normally continue the statement on more than one line.

- To continue statement on multiple lines, either we use constructs with delimiters for the beginning and the end or we use special key - \ - to ignore the enter key – we escape the newline.

  Refer to 3_program_structure.py

```
# 1. use constructs which has beginning and ending markers - like a pair of parentheses
print(
"five"
)
# 2. enter \ (backslash) before pressing <Enter> key - this is called escaping.
print \
(
"six"
```

)

- We can have multiple statements on a single line by using ; as a separator between statements.

  # multiple statements on a single line
  print("seven"); print("eight")

- In many languages, the execution starts from a special function which the programmer has to write called main. We have no such concept in Python. The execution starts from the first statement on top.

- A function name itself has a value. Any thing which has a value is called an expression. If we enter the function name on one line, it becomes a statement. But function will not be called unless we also use a pair of parentheses.

  print # fn name => expr => stmt; no call !!
  ("no output") # string within parentheses is an expr ; no action

concept of constant, variable and the type:

You may remember that you learn arithmetic before moving onto algebra. In arithmetic, we talk about values which do not change. We find the area of a circle of radius 7 units, 14 units and so on. We then learn algebra(monomial, binomial and so on). We learn the identities like (a + b) ^ 2.

Even in programming, we talk about entities which do not change. They are considered as they are. They are said to be constants or literals.
Examples:
3.14 : floating point constant
“pes” : string constant
1729 : integer constant.

Variables have values which can be changed.
Every value has a type. We use the concept of type to classify information. We will discuss about type later.

variable, input and output:
Examine the first version of the file 4_variable_assignment.py.

```
# version 1:
# find the area of a rectangle with given length and breadth
# input() : gets a string from the keyboard
# int() converts a string to an int
l = int(input())
b = int(input())
a = l * b
print("area : ", a)
```

This is the screen shot.
20
10
area : 200

It works. But certain things are not fine. The user does not get to know what he is supposed to key in – int or some string. We should indicate what the user should do. We can specify a string as prompt.
The variable names should be meaningful. Otherwise, when we revisit our program after sometime, we will not know what these stand for. We do not want to call a dog as table, table as pen and so on.

We try the next version – version 2.

```
$ python 4_variable_assignment.py
enter length of a rectangle : 10
enter breadth of a rectangle : 5
area :  50
```

It looks better. Is it not?

But there are a few restrictions in making the variable name.
It can have letters(alphabets) of English, digits and _(underscore). Normally, it should start with a letter of English. There are special words called keywords with specific meaning and those cannot be used as variable – example : for, while.

Valid names:
a1 a_1 a_one
invalid names:
1a for