ignou
THE PEOPLE'S
UNIVERSITY

Indira Gandhi National Open University
School of Computer & Information Sciences

MCSL-205
C and Python Lab

"शिक्षा मानव को बन्धनों से मुक्त करती है और आज के युग में तो यह लोकतंत्र की भावना का आधार भी है। जन्म तथा अन्य कारणों से उत्पन्न जाति एवं वर्गगत विषमताओं को दूर करते हुए मनुष्य को इन सबसे ऊपर उठाती है।"

– इन्दिरा गांधी

"Education is a liberating force, and in our age it is also a democratising force, cutting across the barriers of caste and class, smoothing out inequalities imposed by birth and other circumstances."

– Indira Gandhi

Indira Gandhi National Open University
School of Computer & Information Sciences

MCSL-205
C AND PYTHON LAB

C AND PYTHON LAB

PROGRAMME/COURSE DESIGN COMMITTEE

Prof. (Retd.) S.K. Gupta
IIT, Delhi

Prof. T.V. Vijay Kumar
Dean,
School of Computer & System Sciences,
JNU, New Delhi

Prof. Ela Kumar,
Dean, Computer Science & Engg
IGDTUW, Delhi

Prof. Gayatri Dhingra
GVMITM, Sonipat, Haryana

Mr. Milind Mahajani
Vice President
Impressico Business Solutions
Noida UP

Prof. V.V. Subrahmanyam
Director
SOCIS, IGNOU, New Delhi

Prof P. Venkata Suresh
SOCIS, IGNOU, New Delhi

Dr. Shashi Bhushan
Associate Professor
SOCIS, IGNOU, New Delhi

Shri Akshay Kumar
Associate Professor
SOCIS, IGNOU, New Delhi

Shri M. P. Mishra
Associate Professor
SOCIS, IGNOU, New Delhi

Dr. Sudhansh Sharma
Asst. Professor
SOCIS, IGNOU, New Delhi

BLOCK PREPARATION TEAM

Prof. V. V. Subrahmanyam
SOCIS, IGNOU *(Section-1 Unit Writer)*

Prof S.R.N. Reddy, *(Content Editor)*
HOD, Dept. of CSE,
IGDTUW, Delhi

Dr.Sudhansh Sharma, Asst.Professor
SOCIS, IGNOU *(Section-2 Unit Writer)*

Prof. Parmod Kumar (Language Editor)
SOH, IGNOU, New Delhi

Course Coordinator: Dr. Sudhansh Sharma

PRINT PRODUCTION

Mr. Tilak Raj
Assistant Registrar
MPDD, IGNOU, New Delhi

November, 2021

ISBN: 978-93-5568-130-0

Further information, about the Indira Gandhi National Open University courses may be obtained from the University's office at Maidan Garhi, New Delhi-110 068.

Printed and published on behalf of the Indira Gandhi National Open University by Registrar, MPDD, IGNOU, New Delhi

Laser Composed by Tessa Media & Computers, C-206, Shaheen Bagh, Jamia Nagar, New Delhi

Printed at : Hi-Tech Graphics, D-4/3, Okhla Industrial Area, Phase-II, New Delhi-110020

COURSE INTRODUCTION

This course introduces you to practically work with the two most powerful programming languages i.e. C and Python. After completing this course you will be able to perform programming in both C and Python, the two are chosen to provide you the flavour of both conventional and modern programming. You will appreciate that both programming languages are equally powerful.

To build the carrier path the skill of programming can be a fun and profitable way, but before starting the learning of this skill, one should be clear about the choice of programming language. Before learning any programming language, one should figure out which language suits best to the learner.

This course makes you to compare the functionalities of both C and Python programming language, which may help the learners to analyse and generate a lot of opinions about their choice of programming language.

This course gives you an exposure to both programming languages i.e. C and Python, based on your requirement you can choose your option to build your carrier in programming.

BLOCK INTRODUCTION

This is the lab course, wherein you will have the hands on experience for programming in C and Python. You have studied the concepts of both programming languages i.e. C and Python in support course material (MCS-201 Programming in C and Python). In Section-1, you will apply the concepts of C programming under DOS and LINUX environments, the same are provided there, illustratively. In Section -2, you will apply the concepts ofPython programming under ANACONDA and Google Colabs environments, the same are provided there, illustratively.

A list of programming problems is also provided at the end of each session. Please go through the general guidelines and the program documentation guidelines carefully.

This block consists of 2 Sections and is organized as follows:

Section-1: C Programming Lab

- Comprises of 10 sessions where in 30 programming tasks are framed to be practiced in lab

Section-2: Python Programming Lab

- Comprises of 10 sessions where in 28 programming tasks are framed to be practiced in lab

Happy Programming!!

SECTION 1 C PROGRAMMING LAB

Structure **Page No.**

1.0 INTRODUCTION

This is the lab course, wherein you will have the hands on experience. You have studied the support course material (MCS-201Programming in C and Python). In this section, C programming under DOS and LINUX environments are provided illustratively. A list of programming problems is also provided at the end of each session. Please go through the general guidelines and the program documentation guidelines carefully.

1.1 OBJECTIVES

After completing this lab course you will be able to:

- develop the logic for a given problem ;
- write the algorithm for a given problem;
- draw a flow chart for a given problem;
- recognize and understand the syntax and construction of C code;
- gain experience of procedural language programming;
- know the steps involved in compiling, linking and debugging C code;
- understand how to us header files;
- make use of different data-structures like arrays, pointers, structures and files;
- understand how to access and use library functions;
- understand function declaration and definition;
- feel more confident about writing own functions;
- be able to write some simple output on the screen as well as in the files;
- be able to write some complex programs;
- be able to apply all the concepts that have been covered in the theory course; and
- know the alternative ways of providing solution to a given problem.

1.2 GENERAL GUIDELINES

- You should attempt all problems/assignments given in the list session wise.

- You may seek assistance in doing the lab exercises from the concerned lab instructor. Since the assignments have credits, the lab instructor is obviously not expected to tell you how to solve these, but you may ask questions concerning the C language or a technical problem.

- For each program you should add comments (i.e. text between /* ... */ delimiters) above each function in the code, including the main function. This should also include a description of the function written, the purpose of the function, meaning of the argument used in the function and the meaning of the return value (if any).

These descriptions should be placed in the comment block immediately above the relevant function source code.

- The comment block above the main function should describe the purpose of the program. Proper comments are to be provide where and when necessary in the programming.
- The program written for the problem given should conform to the ANSI standard for the C language.
- The program should be interactive, general and properly documented with real Input/ Output data.
- If two or more submissions from different students appear to be of the same origin (i.e. are variants of essentially the same program), none of them will be counted. You are strongly advised not to copy somebody else's work.
- It is your responsibility to create a separate directory to store all the programs, so that nobody else can read or copy.
- Observation book and Lab record are compulsory.
- The list of the programs(list of programs given at the end, session-wise) is available to you in this lab manual. For each session, you must come prepare with the algorithms and the programs written in the Observation Book. You should utilize the lab hours for executing the programs, testing for various desired outputs and enhancements of the programs.
- As soon as you have finished a lab exercise, contact one of the lab instructor / incharge in order to get the exercise evaluated and also get the signature from him/her on the Observation book.
- Completed lab assignments should be submitted in the form of a Lab Record in which you have to write the algorithm, program code along with comments and output for various inputs given.
- The total no. of lab sessions (3 hours each) are 10 and the list of assignments is provided session-wise. It is important to observe the deadline given for each assignment.

1.3 SALIENT FEATURES OF C

We briefly list some of C's characteristics that define the language and also have lead to its popularity as a programming language. Naturally, we studied many of these aspects throughout the MCS-201 Programming in C and Python course.

- Small size
- Extensive use of function calls
- Structured language
- Low level (bitwise) programming readily available
- Pointer implementation - extensive use of pointers for memory, array, structures and functions.
- It has high-level constructs.
- It can handle low-level activities.
- It produces efficient programs.
- It can be compiled on a variety of computers.

1.4 C PROGRAMMING USING BORLAND COMPILER

C using Borland C/C++ Compiler

Some of you may be using the Borland C/C++ compiler during the lab sessions under MS-DOS connecting through Windows. Whilst C++ is a different programming language to C, it is in fact a superset of C i.e. almost everything that C provides, C++ provides too, and more besides. Therefore we can use Borland C++ to compile our C programs.

To start Borland C/C++

Click the **Start** button in the bottom left hand corner of the screen. The **Start** menu pops up. Select **Programs** from the **Start** menu. Select **Borland C/C++ from** the **Programs** menu. Select **Borland C/C++** from the **Borland C++ menu**. In summary the steps to launching **Borland C/C++** are:

Start--->Programs--->Borland C++--->Borland C++

You should now see the main window for the C/C++ development environment.

Editing a Program

We can create a program by entering text that corresponds to C statements into a file.

Setting Directories

Before you proceed, make sure that the directory settings for Borland C/C++ are correct. This can be done as follows:

Select **Options** from the menu and then select **Project** from the Options pull-down menu. This will display the Project Options dialog box. In the **Topics** area, click on **Directories**. On the right-hand side of the window you will see the **Directories** listed. Ensure that the information in each of the fields is as given below – if it is incorrect, modify it accordingly.

Source Directories
Include c:\ bc5\include
Library c:\ bc5\lib
Source*Leave this field BLANK*
Click on **OK** to continue.

Creating hello.c

- Select **File** from the menu and then select **New** from the file menu. The first thing that you should do is give the program a name, **hello.c**:
- Select **File** from the menu
- Select **Save as** from the File drop-down menu
- In the **Drives** drop-down list box, click on the down arrow to open up the list box.
- Scroll through the list to select the drive and click on it.
- Click on the **File Name:** field and type **hello.c**
- (Make sure the file name has the **.C** extension only. It should not have a **.CPP** extension. If it does not change it to **.C**, or it won't run properly)
- Click on the **OK** button to continue.
- Now type the **hello.c** program exactly as you wrote in the lab observation book.
- Remember that it is good practice to save your programs periodically. You can do this as follows:

- Select **File** from the menu.
- Select **Save** from the File drop-down menu.

Compiling a Program

When you have finished typing in the program, you should compile it as follows:

Select **Project** from the menu
Select **Compile** from the project drop-down menu.

An attempt will be made to compile your program. If there are errors, they will be reported in the message window. You should use the information provided to help you fix the problems and then recompile the program.

Running a Program

If you have successfully compiled your program you can now link and run it as follows:

- Select **Debug** from the menu
- Select **Run** from the Debug drop-down menu.

First, an attempt will be made to link your program. If there are errors, they will be reported in the message window. You should use the information provided to help you fix the problems and then recompile and link the program.

Your program now runs. The output from the program will be displayed in a separate window. (If the screen displays a black output window for a split second and the window then disappears it means you did not set the Target Output type before you compiled your program).

To switch between the edit window and the output window, simply click on the window that you want to activate.

To close the output window, point to the icon in the top left-hand corner and double-click on it.

ALT+F9 (Short cut for Compile and Link)

As you become more familiar with the Borland C/C++ development environment, you will realize that it is possible to combine steps such as **compile** and **link** into a single step such as **build all.** There are also key combinations that can be used instead of selecting from menus (e.g. Alt+F9 compiles the program). You should spend some time familiarizing yourself with the Borland C/C++ development environment.

To close the **hello.c** file, double click on the icon in the top left-hand corner of the **hello.c** edit window.

To Quit from Borland C/C++ compiler

To exit from Borland C/C++:

- Select **File** from the menu
- Select **Exit** from the File drop-down menu

1.5 USING C WITH LINUX

A little knowledge of LINUX operating system and commands is necessary before you can write and compile programs on the LINUX system. Every programmer goes through the same three-step cycle.

1. Writing the program into a file
2. Compiling the program
3. Running the program.

During program development, the programmer may repeat this cycle many times, refining, testing and debugging a program until a satisfactory result is achieved. The UNIX commands for each step are discussed below.

Writing the Program

UNIX expects you to store your program in a file whose name ends in .c. This identifies it as a C program. The easiest way to enter your text is using a text editor like *vi, emacs* or *xedit*. To edit a file called testprog.c using vi type

vi testprog.c

The *editor* is also used to make subsequent changes to the program.

Compiling the Program

There are a number of ways to achieve this, though all of them eventually rely on the compiler (called cc on our system).

The C Compiler (cc)

The simplest method is to type

cc testprog.c

This will try to compile testprog.c, and, if successful, will produce a runnable file called a.out. If you want to give the runnable file a better name you can type

cc testprog.c -o testprog

This will compile testprog.c, creating runnable file testprog.

Running the Program

To run a program under LINUX you simply type in the filename. So to run program testprog, you would type

testprog

or if this fails to work, you could type

./testprog

You will see your prompt again after the program is done.

1.6 PROGRAM DEVELOPMENT LIFE CYCLE

The four steps in the program development life cycle:

- Design algorithm
- Draw flowchart
- Program coding
- Testing for various inputs

Example 1

Represent the complete steps in the program development life cycle that reads the number of letter grades A, B, C, D and F for a student. The program will compute and print the student's grade point average. It should then determine and print the student's academic standing (like high honors, honors, satisfactory, or probation) according to the following table:

Grade Point Average	Academic Standing
3.51 - 4.00	High Honors
3.00 - 3.50	Honors
2.00 - 2.99	Satisfactory
Less than 2.00	Probation

Note: In computing grade point average, assume that the weight of letter grade A is 4, B is 3, C is 2, D is 1 and f is 0.

Step 1: Design the algorithm / pseudocode for the given problem

Print "Please enter your number of subject that your taken last semester"

Set CorrectStatusInput "no"
Print "Enter the number of grade A"
Read number_of_ A
Print "Enter the number of grade B"
Read number_ of_B
Print "Enter the number of grade C"
Read number_of_C
Print "Enter the number of grade D"
Read number_of_D
Print "Enter the number of grade F"
Read number_of_F
Compute Total_subject = number of grade A + number of grade B + number of grade C + number of grade D + number of grade F
while CorrectStatusInput "no" and Total_subject < =0
begin
print "Enter number of grade A"
read number of A
print "Enter number of grade B"
read number of B
print "Enter number of grade C"
read number of C
print "Enter number of grade D"
read number of D
print "Enter number of grade F"
read number of F

if (**number_of_A** greater than and equal 0 and **number_of_B** greater than and equal 0 and **number_of_C** greater than and equal 0 and **number_of_ D** greater than and equal 0 and **number_of_F** greater than 0) and (Total_subject >0)

set CorrectStatusInput "yes"

end if
end
end while

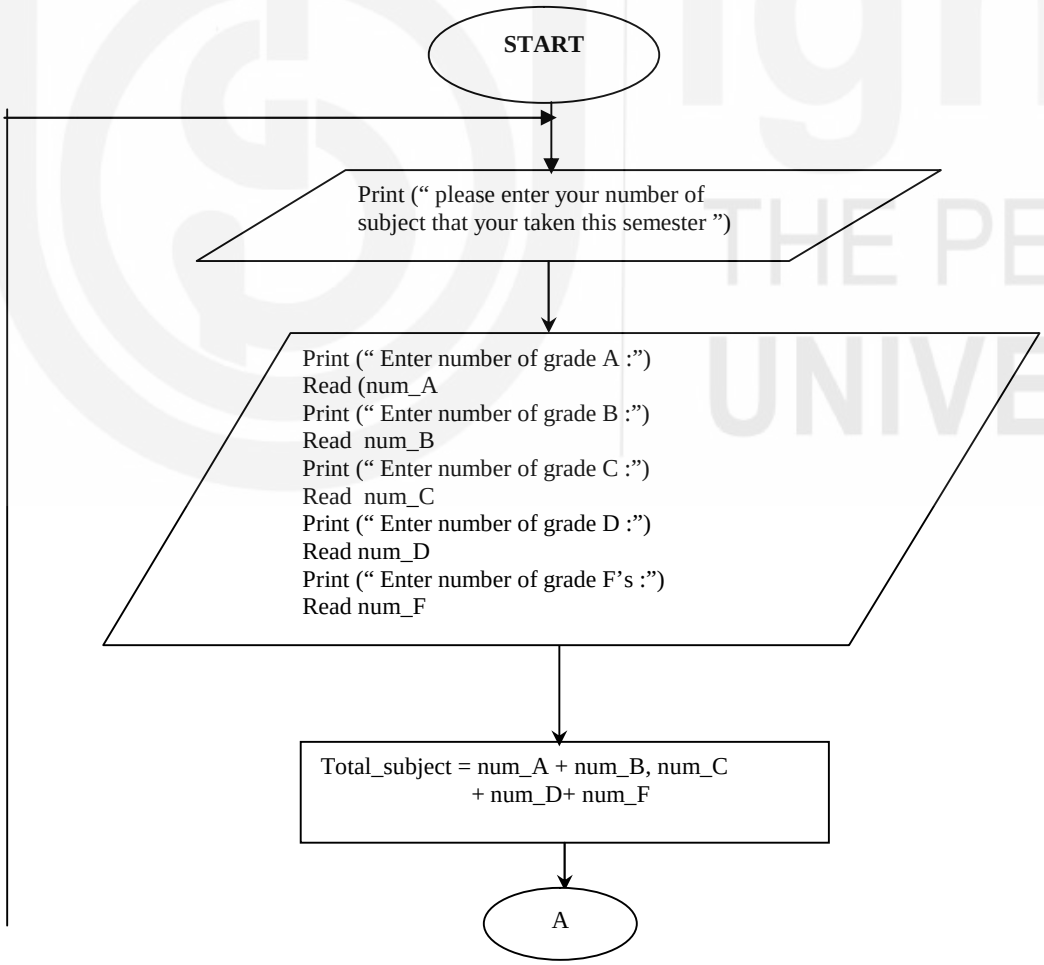
compute Total_point = **number of grade A * 4 + number of grade B *3 +**

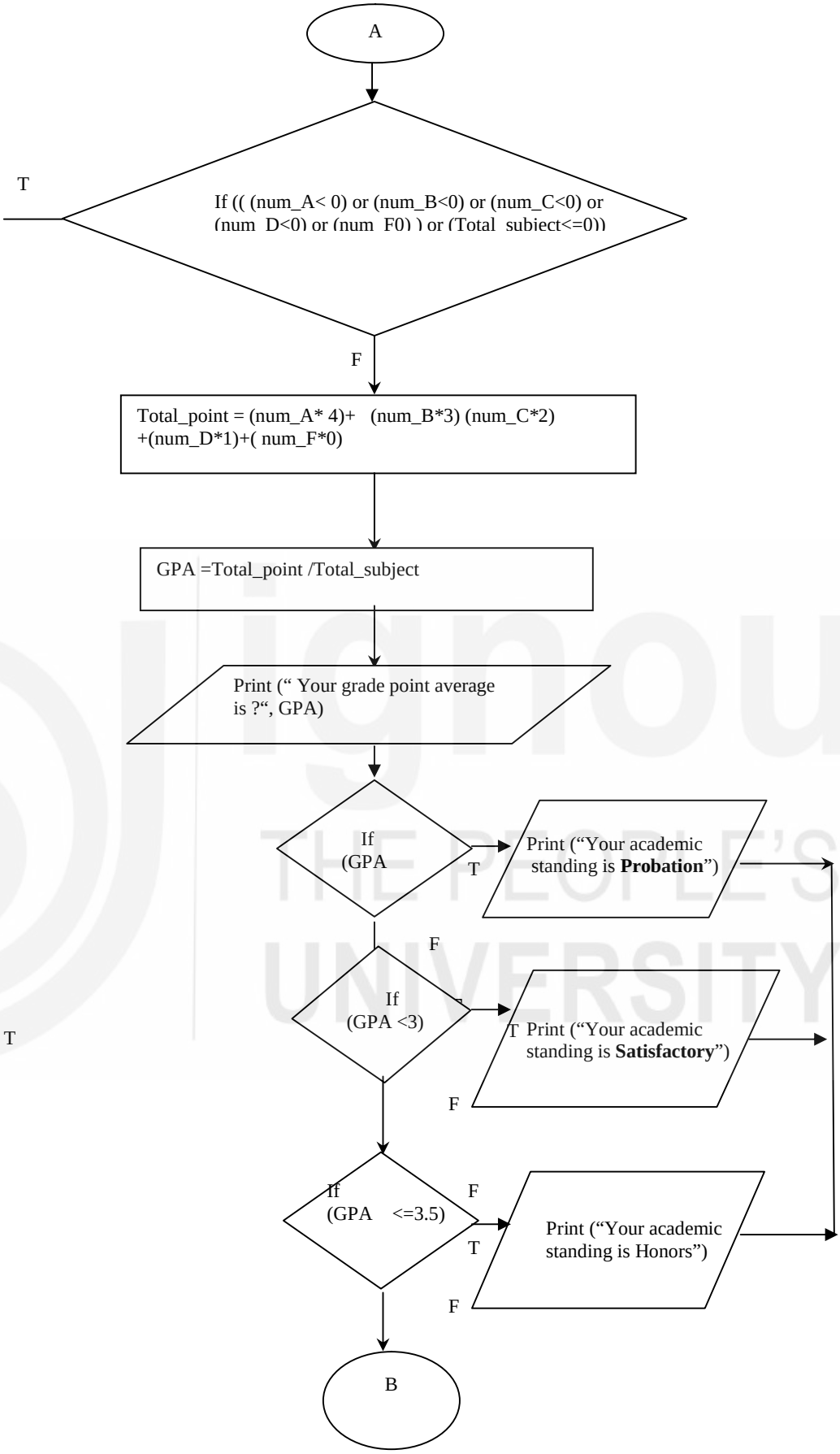
number of grade C*2 + number of grade D *1 +
number of grade F *0

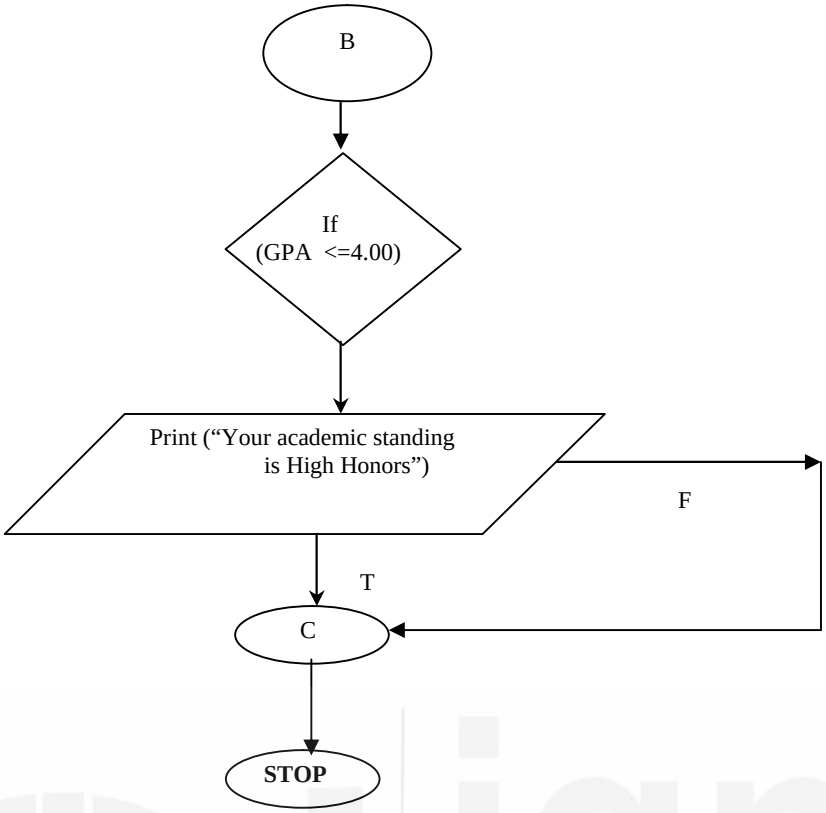
compute Total_average = **Total point / Total subject**

```
print  Total_average
if  Total_ average less than 2
        print " Your academic standing is Probation"
    else
        if Total_average less than 3
            print " Your academic standing is Satisfactory"
          else
            if  Total_average less than or equal 3.5
                print " Your academic standing is Honors"
              else
                if Total_average less than or equal 4.00
                    print " Your academic standing is High Honors"
                 end_if
            end_if
      end_if
end_if
```

Step 2:Design a flowchart based on the algorithm.




A
T
If (((num_A< 0) or (num_B<0) or (num_C<0) or
(num_D<0) or (num_F0)) or (Total_subject<=0))
F
Total_point = (num_A* 4)+ (num_B*3) (num_C*2)
+(num_D*1)+(num_F*0)
GPA =Total_point /Total_subject
Print (" Your grade point average
is ?", GPA)
If
(GPA
T
Print ("Your academic
standing is **Probation**")
F
If
(GPA <3)
T
T Print ("Your academic
standing is **Satisfactory**")
F
If
(GPA <=3.5)
F
T
Print ("Your academic
standing is Honors")
F
B



Step 3: Translate the flowchart to C source code.

```
/* Program to computer and print the grade point average */

#include <stdio.h>

main()

{

        int num_A,num_B,num_C,num_D,num_F;     /* Variables declaration part*/
        int total_subject;
        int total_point ;
        float GPA;


        do {
                printf(" \tThis is for calculate your GPA \n");
                printf("Enter number of grade A : ");
                scanf ("%d", &num_A);

                printf("Enter number of grade B : ");
                scanf ("%d", &num_B);

                printf("Enter number of grade C : ");
                scanf ("%d", &num_C);

                printf("Enter number of grade D : ");
                scanf ("%d", &num_D);

                printf("Enter number of grade F : ");
                scanf ("%d", &num_F);

                /* calculation of the total */
```

```
                total_subject = num_A + num_B + num_C + num_D + num_F;

{
        if (((num_A <0) || (num_B <0)||(num_C<0)||(num_D <0)||(num_F
<0))||(total_subject <=0))
 printf (" you should enter your number that you having take again\n");
                }
} while (((num_A <0) || (num_B <0)||(num_C <0)||(num_D <0)||(num_F
<0))||(total_subject <=0));

/* this to calculate and print total_subject, total_point and GPA */

total_point = num_A *4 + num_B*3 + num_C*2 + num_D*1 + num_F*0;


GPA =(float) (total_point / total_subject) ;

/* this to print total subject */

printf ("\nYour grade point average is %.2f \n",GPA);


        /* this selection to get output the students status */

 if (GPA <2)
 printf ("Your academic standing is Probation\n\n");
   else
        if (GPA <3)
           printf ("Your academic standing is Satisfactory\n\n");
             else
                  if (GPA <= 3.5)
                    printf ("Your academic standing is Honors\n\n");
                      else
                           if (GPA <= 4.00)
                           printf    ("Your    academic    standing    is    High
honors\n\n");

        return 0;
}
```

Step 4: Testing

OUTPUT

```
This is for calculate your GPA
Enter number of grade A: 3
Enter number of grade B: 2
Enter number of grade C: 2
Enter number of grade D: 0
Enter number of grade F: 0
Your grade point average is 3.14.
Your academic standing is Honors.
```

1.8 LIST OF LAB ASSIGNMENTS – SESSIONWISE

Session 1:

1. Develop an algorithm and draw corresponding flowchart and also write an interactive C program for the following:
   (a) To check if a given integer is odd or even.
   (b) To calculate the sum of odd and even numbers from a given list of numbers.
   (c) To check whether the given integer is positive or negative.
   (d) To find whether the given integer is divisible by 7.
   (e) To compute and display the sum of digits of a given 5 digit number.

Session 2:

2. Design a flow chart and write an interactive program for the problem given below:
   (a) To convert the given decimal number to binary
   (b) To convert the octal number to decimal

3. Design a flowchart and write an interactive program that reads in integers until a 0 is entered. If it encounters 0 as input, then it should display:

   - Number of even and odd integers
   - Sum and average value of even integers
   - Sum and average value of odd integers.

   Note: Use *switch* statement for selection.

4. Write an interactive program to generate the divisors of a given integer.

Session 3:

5. Write a program to find all Armstrong number in the range of 0 and 999
   *Hint: An Armstrong number of three digits is an integer such that the sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since 3**3 + 7**3 + 1**3 = 371.*

6. Write a program to check whether a given number is a perfect number or not.
   Hint: *A positive integer n is called a* ***perfect number*** *if it is equal to the sum of all of its positive divisors, excluding n itself. For example, 6 is a perfect number, because 1, 2 and 3 are its proper positive divisors and 1 + 2 + 3 = 6. The next perfect number is 28 = 1 + 2 + 4 + 7 + 14. The next perfect numbers are 496 and 8128.*

7. Write a program to check whether given two numbers are amicable numbers or not.
   Hint: Amicable numbers *are two numbers so related that the sum of the proper divisors of the one is equal to the other, unity being considered as a proper divisor but not the number itself. Such a pair is (220,284); for the proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110, of which the sum is 284; and the proper divisors of 284 are 1, 2, 4, 71, and 142, of which the sum is 220.*

8. Write a program to find the roots of a quadratic equation.

Session 4:

9. Write an interactive C program to find the sum and average of a given array of elements.

10. Write an interactive C program to find the second largest and smallest elements in a given single dimensional array.

11. Write a program to find the largest number in an array.

12. Write a program in C to put the given list of odd and even numbers in two different arrays (Even_Array and Odd_Array).

Session 5:

13. Write an interactive program to do the following computations by providing the option using the *switch* statement:

    - Add two matrices
    - Subtract two matrices
    - Multiply two matrices

14. Write a program to compute transpose of a matrix.

15. Write a program to find the inverse of a matrix.

Session 6:

16. Using function, swap the values of 2 variables.

17. Using function, write a C program to display the prime numbers between intervals.

18. Write a program to check whether the number is a Prime or Armstrong number using user defined function.

19. Using user defined function, check whether the given year is a leap year or not.

Session 7:

20. Using recursion write a C program,
    (a) To find sum of digits of a given number.
    (b) To reverse a given number
    (c) To find the factorial of a number
    (d) To find Greatest Common Divisor (GCD) of two numbers
    (e) To generate Fibonacci sequence

Session 8:

21. Write a program to convert a given lowercase string to upper case string without using the inbuilt string function.

22. Write a program to count number of vowels, consonants and spaces in a given string.

23. Write a program to input a string and output the reversed string, i.e. if "USF" is input, the program has to output "FSU". You are **not** to use array notation to access the characters, instead use pointer notation.

24. Write a program to find the length of the string without using the inbuilt string function, instead use pointer notation.

Session 9:

25. Write a program to process the students-evaluation records using structures.

26. Using structures, write a program to process the electricity bill.

27. Using structures, write an interactive C program to process the retail medical stores bill.

Note: Assumptions can be made wherever necessary for Q25 to Q27.

Session 10:

28. Write a program that prompts the user the name of a file and then counts and displays the number of bytes in the file. And create a duplicate file with the word '.backup' appended to the file name. Please check whether file was successfully opened, and display an error message, if not.

29. Write a program to create a file, open it, type-in some characters and count the number of characters in a file.

30. Write a program that will input a person's first name, last name, Employee number and age and write the information to a data file. Each person's information should be in a single line. Use the function fprintf to write to the data file. Accept the information and write the data within a loop. Your program should exit the loop when the word "EXIT" is entered for the first name. Remember to close the file before terminating the program..

SECTION 2 PYTHON PROGRAMMING LAB

Structure **Page o. N**

2.0 INTRODUCTION

This is the lab course, wherein you will have the hands on experience. You have studied the support course material (MCS-201 Programming in C & Python). In this part, Python programming under ANACONDA and Google Colabs environments are provided illustratively. A list of programming problems is also provided at the end of each session. Please go through the general guidelines and the program documentation guidelines carefully.

2.1 OBJECTIVES

After completing this lab course you will be able to:

- develop the logic for a given problem;
- recognize and understand the syntax and construction of Python code;
- gain experience of Python as Object Oriented Programming Language;
- know the steps involved in executing the python code;
- •understand how to access and use modules;
- understand function declaration and definition;
- feel more confident about writing your own functions;
- be able to write some simple output on the screen as well as in the files;
- be able to write some complex programs;
- be able to apply all the concepts that have been covered in the theory course; and
- know the alternative ways of providing solution to a given problem.

2.2 GENERAL GUIDELINES

- You should attempt all problems/assignments given in the list session wise.
- You may seek assistance in doing the lab exercises from the concerned lab instructor. Since the assignments have credits, the lab instructor is obviously not expected to tell you how to solve these, but you may ask questions concerning the Python language or a technical problem.

- For each program you should add comments above each function in the code, including the main function. This should also include a description of the function written, the purpose of the function, meaning of the argument used in the function and the meaning of the return value (if any).
- These descriptions should be placed in the comment block immediately above the relevant function source code.
- The comment block above the main function should describe the purpose of the program. Proper comments are to be provide where and when necessary in the programming.
- The program should be interactive, general and properly documented with real Input/ Output data.
- If two or more submissions from different students appear to be of the same origin (i.e. are variants of essentially the same program), none of them will be counted. You are strongly advised not to copy somebody else's work.
- It is your responsibility to create a separate directory to store all the programs, so that nobody else can read or copy.
- Observation book and Lab record are compulsory
- The list of the programs(list of programs given at the end, session-wise) is available to you in this lab manual. For each session, you must come prepare with the algorithms and the programs written in the Observation Book. You should utilize the lab hours for executing the programs, testing for various desired outputs and enhancements of the programs.
- As soon as you have finished a lab exercise, contact one of the lab instructor / in-charge in order to get the exercise evaluated and also get the signature from him/her on the Observation book.
- Completed lab assignments should be submitted in the form of a Lab Record in which you have to write the algorithm, program code along with comments and output for various inputs given.
- The total no. of lab sessions (3 hours each) are 10 and the list of assignments is provided session-wise. It is important to observe the deadline given for each assignment.

2.3 SALIENT FEATURE OF PYTHON

We learned from the comparison of C and Python, given in MCS - Programming in C & Python, that Python is a high-level, general-purpose, interpreted high level programming language. It is dynamically typed and garbage-collected, and supports multiple programming paradigms like structured (particularly, procedural,) object-oriented, and functional programming, and due to its comprehensive standard library Python is often described as a "batteries included" language

In this course you are given exposure to both programming languages i.e. C and Python, based on your requirement you can choose your option to build your carrier in programming. A comparison among the salient features of both languages is given below:

Metrics	Python	C
Introduction	It is a high-level, general-purpose & interpreted programming language.	C is general-purpose procedural programming language.
Speed	Being Interpreted programming language its execution speed is slower then that of the compiled programming language (i.e. C).	Being compiled programming language its execution speed is faster then that of the interpreted programming language (i.e. Python).
Usage	Number of lines of code written in Python is quite less in comparison to C	Program syntax of C is quite complicated in comparison to Python.
Declaration of variables	Declaration of Variable type is not required, they are un-typed and a given variable can be stuck different types of values at different instances during the execution of any python program.	In C, declaration of variable type is must, and it is done at the time of its creation, and only values of that declared type must be assigned to the variables.
Error Debugging	Error debugging is simple, as it takes only one instruction at a time. Compilation and execution is performed simultaneously. Errors are instantly shown, and execution stops at that instruction only.	Being compiler dependent language, error debugging is difficult in C i.e. it takes the entire source code, compiles it and finally all errors are shown.
Function renaming mechanism	the same function can be used by two different names i.e. it supports the mechanism of function renaming	Function renaming mechanism is not supported by C i.e. the same function cannot be used by two different names.
Complexity	Syntax of Python programs is Quite simple, easy to read, write and learn	The syntax of a C program is comparatively harder than the syntax of Python.
Memory-management	It supports automatic garbage collection for memory management.	In C, the Programmer has to explicitly do the memory management.
Applications	Python is a General-Purpose programming language can be used for web design, AI, ML etc.	C is generally used for hardware related applications where you need efficiency.
Built-in functions	Library of built-in functions is quite large in Python.	Library of built-in functions is quite limited in C
Implementing Data Structures	Gives ease of implementing data structures with built-in insert, append functions.	Explicit implementation of functions is requited for the implementation of data structures
Pointers	Pointers functionality is not available in Python.	Pointers are frequently used in C.

2.4 WORKING WITH PYTHON

Now, we are having a bit of clarity about the Python as a programming language, from the past sections we learned about the various libraries and frameworks of Python. Now, we need to work on the IDEs (Integrated Development Environments) of Python, there are many IDEs like Jupyter Notebook, Spyder, VS Code, R Programming etc., all are collectively available in Anaconda (Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.)), or you may also go for the cloud versions Like Google Colab Notebook, where you need not to have high configuration hardwares, only internet is required and through your gmail account you can work on Python using Jupyter notebook.

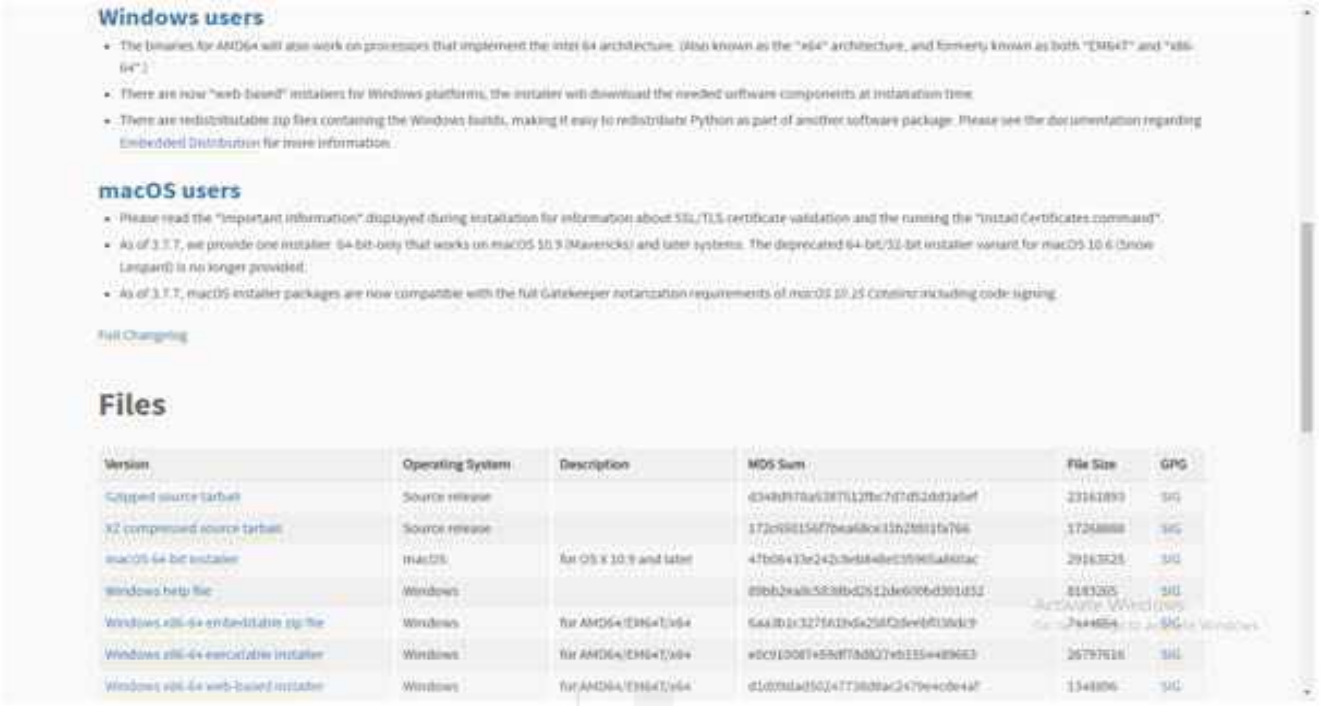
We will discuss in brief, some of the ways to work with Python, you may choose any or try all and other options too.

1) Just browse for https://www.python.org and perform following steps :

   a) Download the latest version of Python for the operating system installed on your computer, as in my case its windows 64 bit, so I downloaded python-3.7.7 (python-3.7.7-amd64.exe) from https://www.python.org/downloads/release/python-377/

   b) Now Run this exe file and install the Python, just click next and go ahead, till the setup installation is finished

   c) Finally you are having an interface for Python programming



There is a variety of IDEs for python, Like Jupyter Notebook, Spyder, VS Code etc, and all are available at Anaconda (a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.)). To start with Anaconda Just perform following steps.

2) Just browse https://www.anaconda.com/ and perform following steps:

   a) Click the Download button on the webpage of https://www.anaconda.com/

   b) the distribution section https://www.anaconda.com/distribution/ will open,

   c) click the download option given on this page https://www.anaconda.com/distribution/

   d) https://www.anaconda.com/distribution/#download-section will open, the option of 64bit and 32bit graphic installer for Python 3.x (currently 3.7) and 2.x (currently 2.7) are given under Anaconda 2020.02 for windows installer.

   e) It is recommended to download 64bit version of Python 3.x (currently 3.7),

   f) Anaconda3-2020.02-Windows-x86_64.exe will be downloaded.

   g) Now, just run the setup of this Anaconda3-2020.02-Windows-x86_64.exe , and click next-next, till the installation is completed.

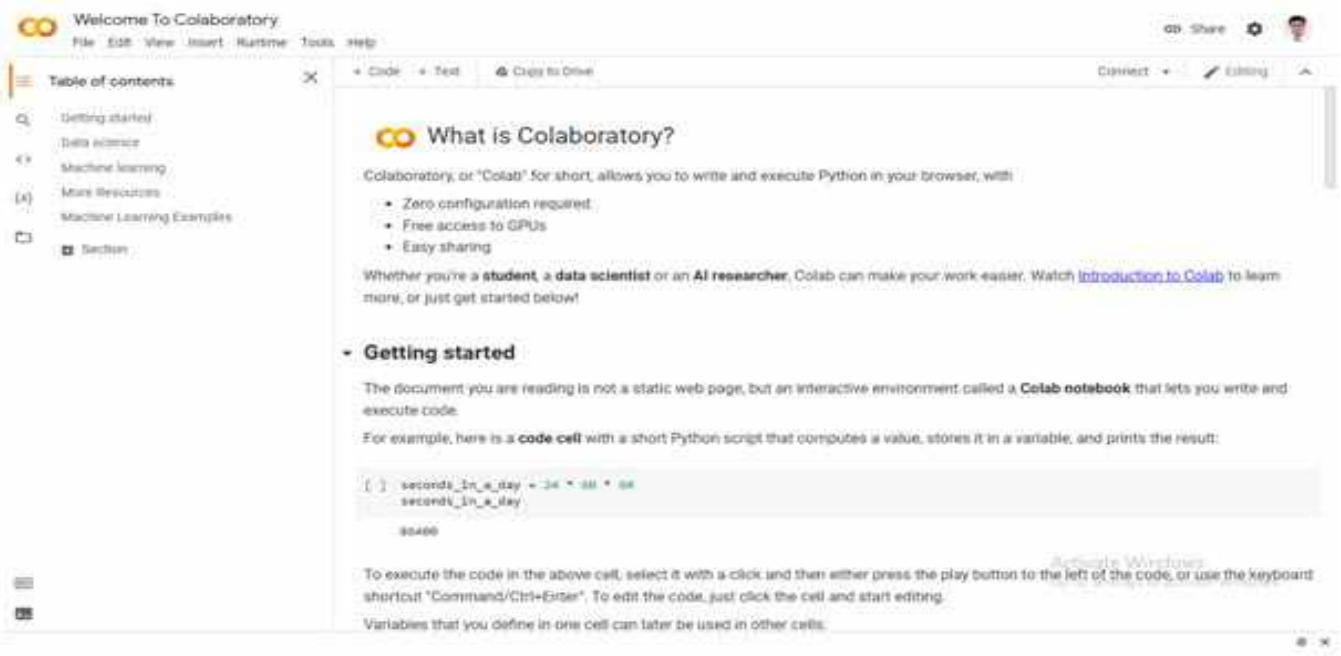
   h) Finally, you will find Anaconda Navigators shortcut on your desktop, click on it and you can start working with any IDE be it Jupyter Notebook, Spyder, VS Code etc., even you can work with R-Programming.

Important: Before working with IDEs you need to understand how to work with them and which ine is suitable, following are observations, currently:

a) When you start Jupyter Notebook on windows (by clicking on the jupyter section, given in the anaconda Navigator), a browser will open in internet explorer, many a times Jupyter Notebook won't work here, then just copy the URL from the Internet Explorer and paste it in the Google Chrome, you will find that Jupyter Notebook starts working, other details regarding the writing, saving, execution of program, will be discussed later.

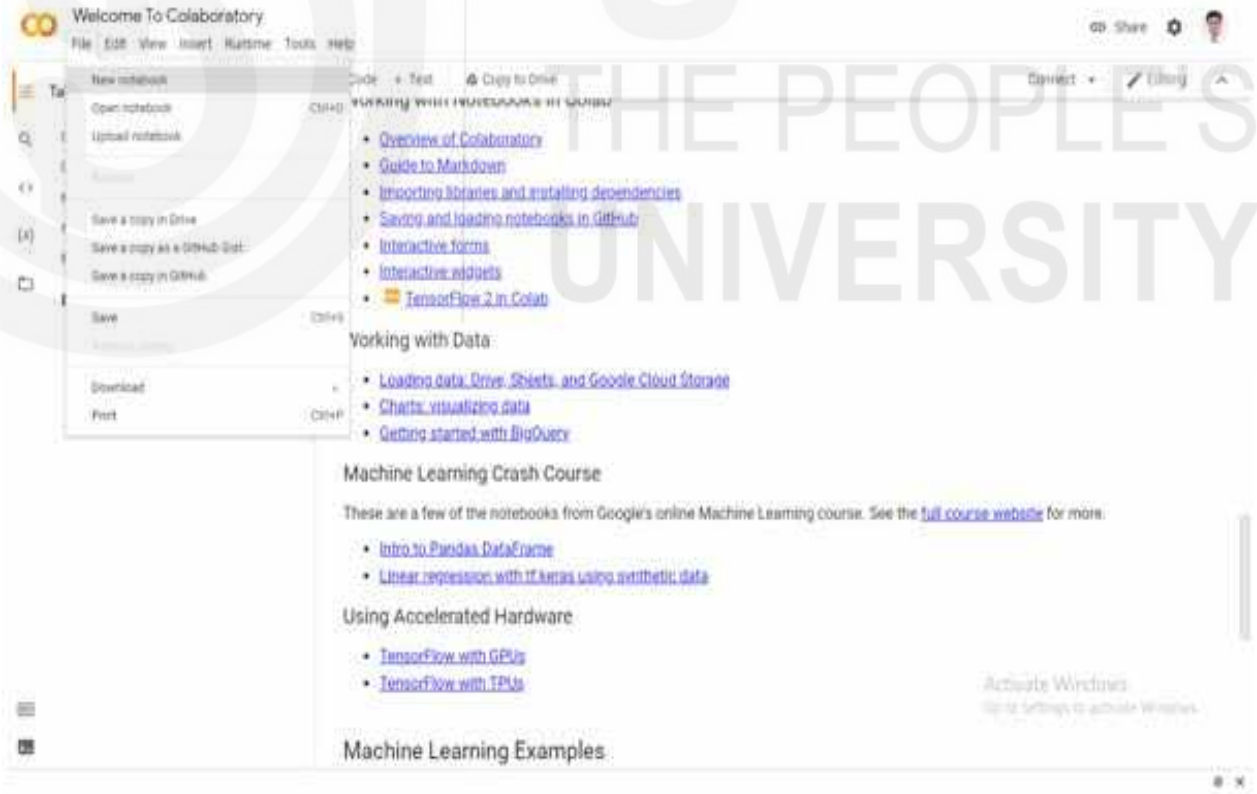
b) Program exeution in Jupyter is line by line and in VS Code and Spyder it goes sideways, even errors can also be seen sideways. Any ways all are good to work with Python.

3) Many a times the learners may not be equipped with the systems having latest hardware configuarations, as desired for the installation of Python, or their might be compatibility issues with operating syatem or may be due to any reason you are not able to install and start your work with Python. Under such circumstances the solution is Google Colab Notebook (https://colab.research.google.com /notebooks/ welcome.ipynb), use this and just login with your gmail account and start your work on Jupyter Notebook, as simple as that.

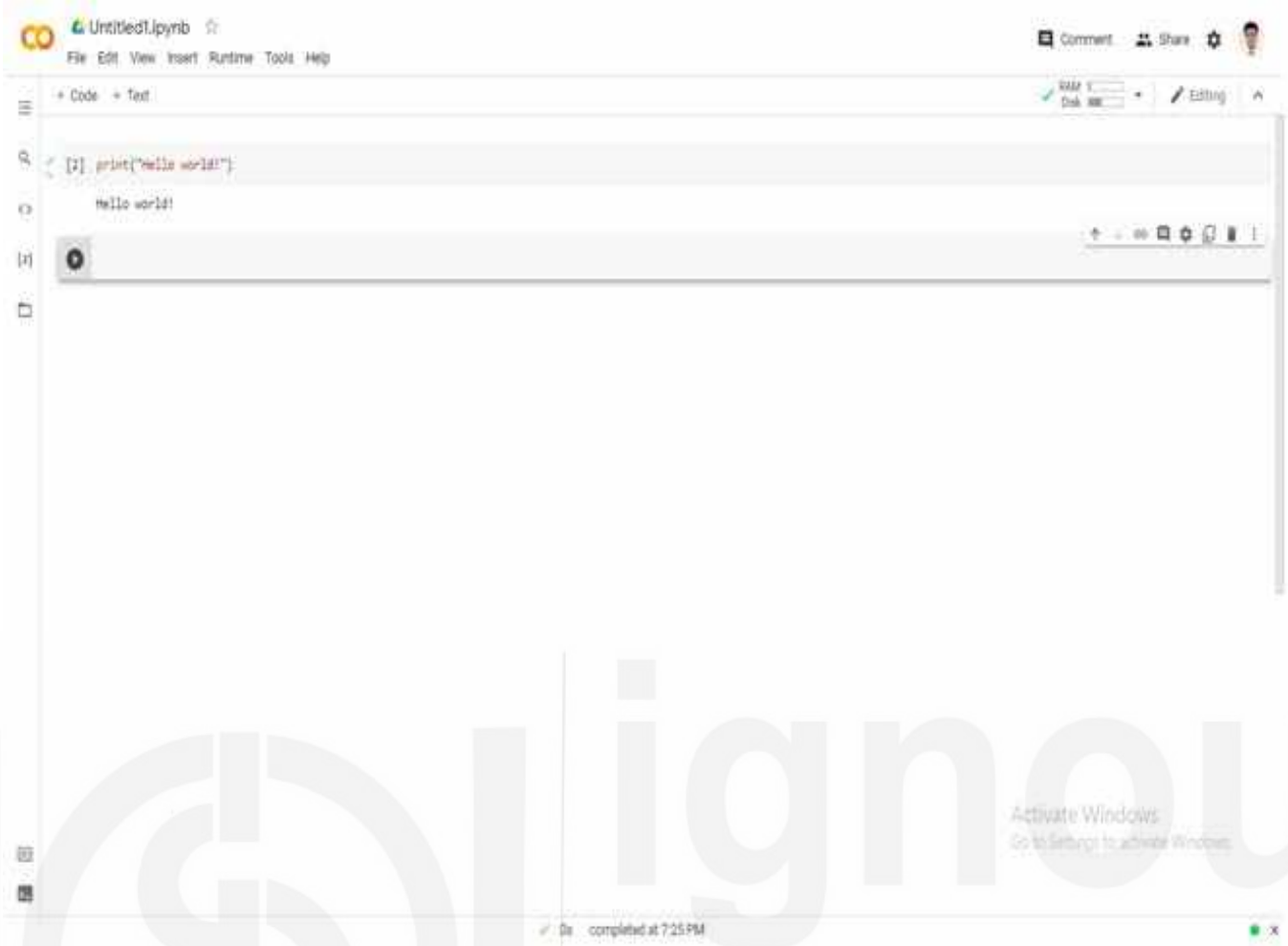


2.5 RUNNING PYTHON PROGRAMS

Just click file option and select new workbook, and new Jupyter notebook will open in Google Colab, now you may start your work



As here, I wrote my first program (print(“hello”)) to say “hello” to all of you , and executed it by simply pressing the play button, you may see just before the print command I wrote, and the output comes just beneath the statement print(“hello”).




```
print('Hello world!')
x = 2020
print(x)

output
Hello World!
2020
```

2.6 LIST OF LAB ASSIGNMENTS – SESSION WISE

Let us try solving a different kind of a problem now. How do you solve an equation in Python?

Session-1

1. Write a program in python, that calculates thevolume of a sphere (given as $\frac{4}{3}\pi r^{3.}$), with radius r entered by the user. (Assume $\pi = 3.14$)
2. Write a program in python that prints a warning message if the weight of a customer's luggage exceeds the allowed limit of 25 kgs.
3. let us say a teacher decided to give grades to her students as follows:

a) Mark greater than or equal to 80 *Excellent*

b) Mark greater than or equal to 65 but less than 80 *Good*

c) Mark greater than or equal to 50 but less than 65 *Pass*

d) Mark less than 50 *Fail*

Write a program in python to print a grade according to a student's mark with multiple if statements

Session - 2

4. Write a program in python, to print even numbers in range 0 to 9 (last number not less than 10).Using while construct
5. Write a program in Python that prints the square of the numbers in the list, using for loop construct.
6. Write a program in Python that will print numbers from 1 to 10 only if the number is odd. When the number is even, it goes to the next iteration. When n becomes 7 it will finish the current while statement and run the next statement which is print (‘all numbers printed').

Session - 3

7. Write a program in Python to perform the Simple slicing on the list ['a','b','c','d','e','f','g','h','i','j','k','l','m','n'], and extracts the characters from elements at index 3 to element at 7 (i.e. upper index - 1). Remember that indexing starts with 0.
8. Write a program in Python to Swap Two Numbers without Using Intermediate/Temporary Variable. Prompt the User for Input.
9. Write a program in Python that Accepts a Sentence and Calculate the Number of Digits, Uppercase and Lowercase Letters.

Session - 4

10. Write a program in Python to check if a given year is a leap year
11. Write a Program to Find

a) the Transpose of a Matrix

b) product of two matrices

c) sum of two matrices

12. Write a program in Python to display the Fibonacci Sequences up to nth term where n is provided by the user

Session - 5

13. Write a function in Python to find the factorial of a number(n), where n is provided by the user
14. Write a function in Python to calculate sum, difference, product of the two numbers passed as an argument to the function.

15. Write a function in Python which will accept a number within the range of 1 and 20, and find whether it is a prime number or not. If it is a prime number, function should return True and if not False.

    Note that prime numbers are numbers which are divisible only by 1 and by the number it-self. What is the argument type that you use here?

Session - 6

16. Write Lambda function and Normal function in Python to get the average of two numbers.
17. Write a Python program to sort a list of tuples using Lambda.
18. Write program in Python to perform following file operations
    a) display file contents
    b) Create file and add contents to it.
    c) read, write and append using *with* statement
    d) to copy a file from 'original.txt' to 'duplicate.txt'
    e) to delete a file

Session - 7

19. Develop a module named calculator.py in Python, which contain three function – factorial(n) : to find factorial of a number, Fibonacci(n): to find Fibonacci series up to given number, n. Perform following tasks:
    a) Import the developed module
    b) call the functions available in the module
        i) to generate Fibonacci series upto a number, and
        ii) to determine factorial of a number entered by user
20. Create a package in Python name Interest and within this package create two modules simple.py and compound.py. The simple.py module contains function Simple_Interest() and compound.py module contains function Compound_Interest().

Session - 8

21. Write Python class to :
    a) to convert an integer to a roman numeral
    b) to convert a roman numeral to an integer.
22. Write a Python class which has two methods get_String and print_String. get_String accept a string from the user and print_String print the string in upper case
23. Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.

Session - 9

24. Write a python code to read a dataset (may be CSV file) and print all features i.e. columns of the dataset. Determine the descriptive statistics i.e. Maximum, Minimum Mean Median, Count, Variance,Standard Deviation etc. of the numeric features like age, salary etc., may be present in the dataset.

25. Write a Python program to read a given CSV file

    a) having tab delimiter.

    b) as a list.

    c) as a dictionary.

    d) print the content of the columns

26. Write a Python program to write a Python dictionary to a csv file. After writing the CSV file read the CSV file and display the content.

27. Write a Python program to write a Python list of lists to a csv file. After writing the CSV file read the CSV file and display the content.

Session -10

28. Write Python Program to perform following tasks

    a) Create a database TRAINING_DB

    b) Set connection with mysql.connector.connect.

    c) Create a table EMP_TRAINING in database TRAINING_DB with following data FIRST_NAME,LAST_NAME,AGE,GENDER,INCOME.

    d) change table structure / (add, edit, remove column of a table) at run time

        i) add a column address in the EMP_TRAINING table.

        ii) execute SQL *INSERT* statement to create a record into EMP_TRAINING table

        iii) run the query to updates all the records having GENDER as '**M**', and increase AGE of all the males by one year.

        iv) delete all the records from EMP_TRAINING Table where AGE is less than 18 .

QR Code -website ignou.ac.in

QR Code -e Content-App

QR Code - IGNOU-Facebook
(@OfficialPageIGNOU)

QR Code Twitter Handel
(OfficialIGNOU)

INSTAGRAM
(Official Page IGNOU)

QR Code -e Gyankosh-site

IGNOU SOCIAL MEDIA

QR Code generated for quick access by Students

IGNOU website

eGyankosh

e-Content APP

Facebook (@offical Page IGNOU)

Twitter (@ Offical IGNOU)

Instagram (official page ignou)

Like us, follow-us on the University Facebook Page, Twitter Handle and Instagram

To get regular updates on Placement Drives, Admissions, Examinations etc.

MPDD/IGNOU/P.O. 2K/November, 2021

ISBN: 978-93-5568-130-0