



Mawlana Bhashani Science and Technology University

Lab-Report

Report No:01

Course code: ICT-3208

Course title: Network Planning and Designing Lab

Date of Performance:06.01.2021

Date of Submission:08.01.2021

Submitted By

Name:Kallol Devnath

ID:IT-18029

Name:Nandon Kumar Pal

ID:IT-18022

3rd year 2nd semester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

1. Objectives

The objective of the lab 1 is to:

- Setup python environment for programing,
- Learn the basics of python,
- Create and run basic examples using python

2. Theory

The official definition of Python is:

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to objectoriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

Main Features of Python:

The main features of Python are:

- **Simple:** Python is a simple and minimalistic language. This pseudo-code nature of Python is one of its greatest strengths.
- **Easy to Learn:** Python is extremely easy to get started with. Python has an extraordinarily simple syntax.
- **Free and Open Source:** Python is an example of FLOSS (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read it's source code, make changes to it, use pieces of it in new free programs, and that you know you can do these things. FLOSS is based on the concept of a community which shares knowledge.
- **High-level Language:** When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

- **Portable:** Due to its open-source nature, Python has been ported (i.e. changed to make it work on) to many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.
- **Multi-Platform:** Python can be used on Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and even PocketPC.
- **Interpreted:** Python does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called byte codes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!
- **Object Oriented:** Python supports procedure-oriented programming as well as object oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality.
- **Extensible:** If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use them from your Python program.
- **Embeddable:** You can embed Python within your C/C++ programs to give 'scripting' capabilities for your program's users.
- **Extensive Libraries:** The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, ftp, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), Tk, and other system-dependent stuff. Remember, all this is always available wherever Python is installed.

3. Methodology Section

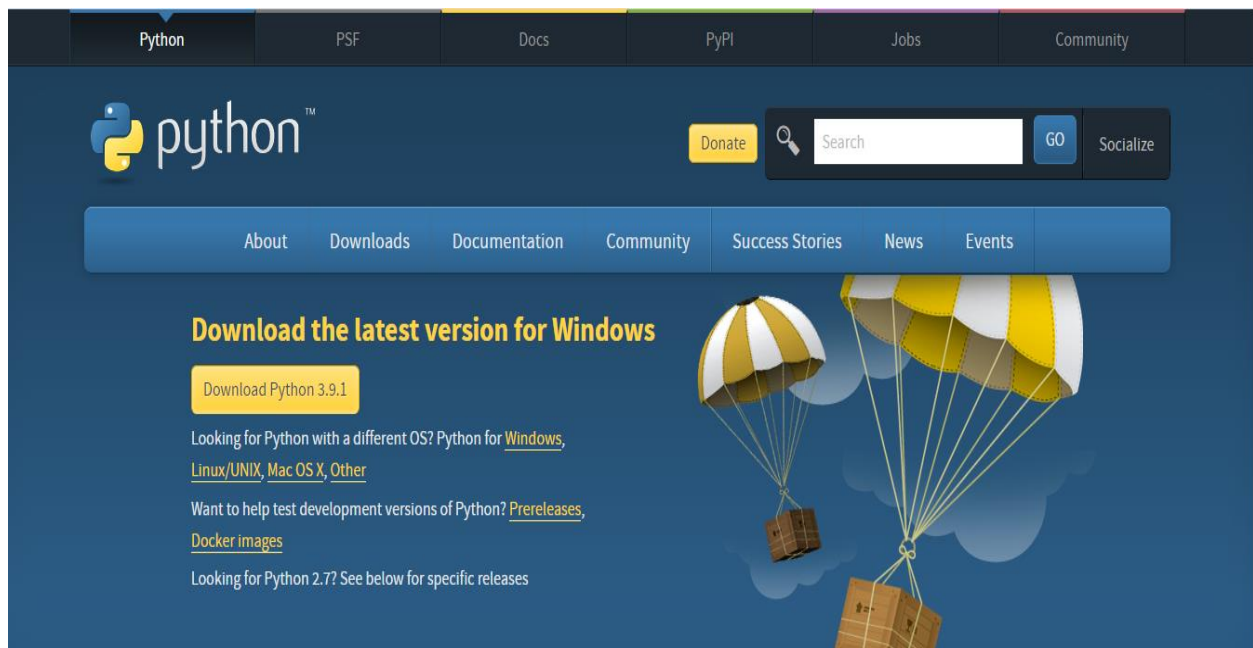
Section 3.1:

Download python3 and Pycharm IDE and install them.

STEP 1:

In order to set up follow the instructions :

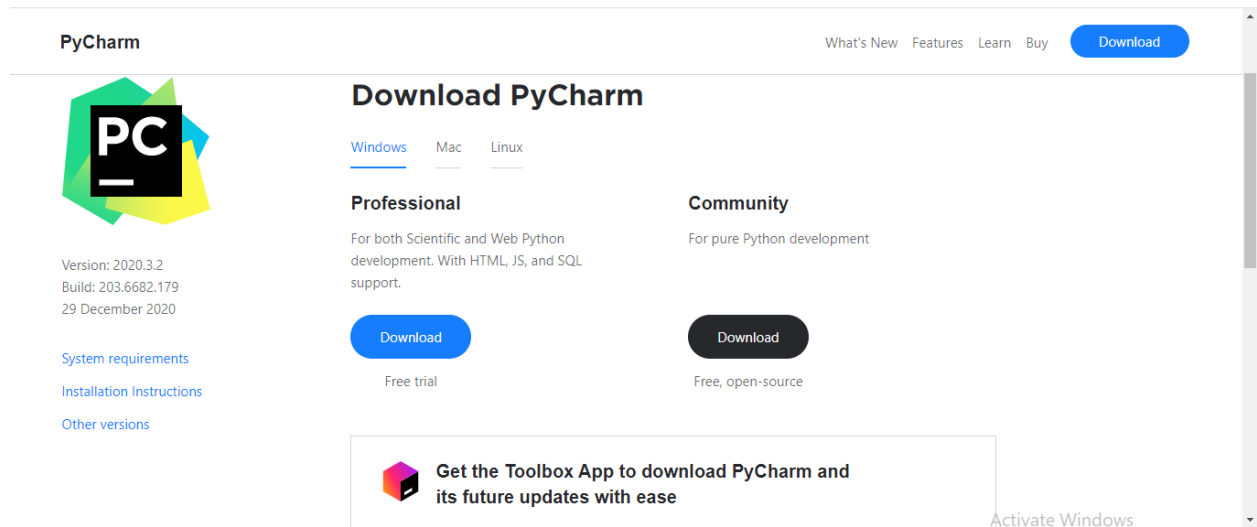
a.Go to <https://www.python.org/downloads/>



b.Download the latest version for windows

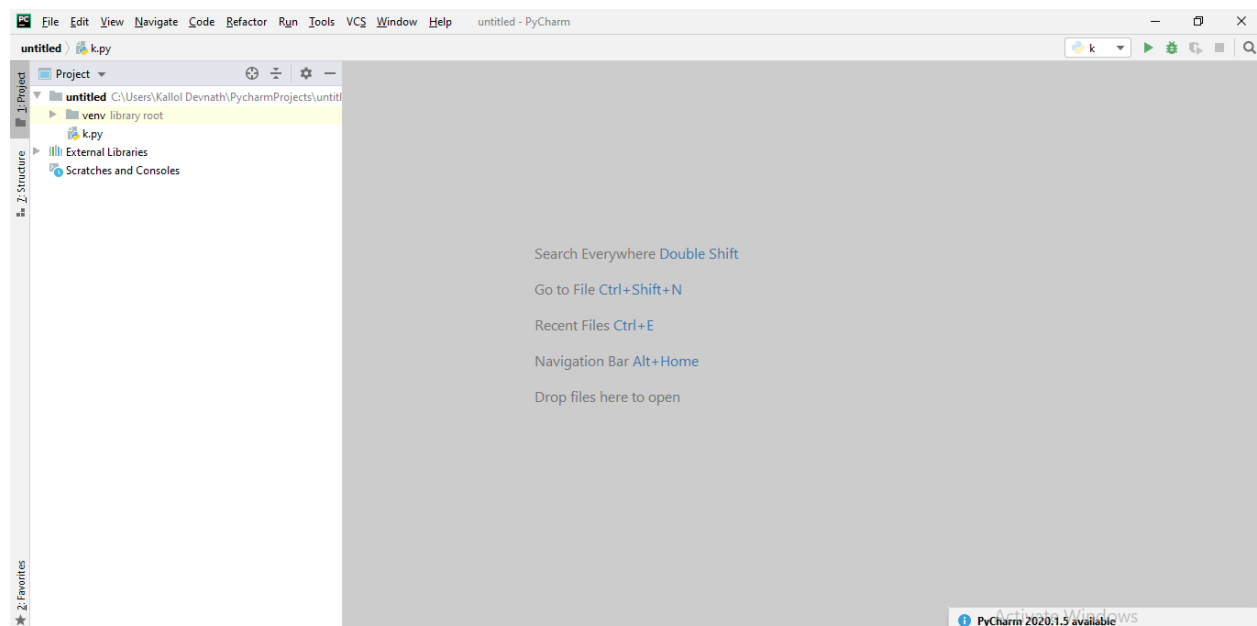
c. Go to <https://www.jetbrains.com/pycharm/download/#section=windows>

d. Download Pycharm IDE Professional



e. Install python3

f. Start pycharm IDE.

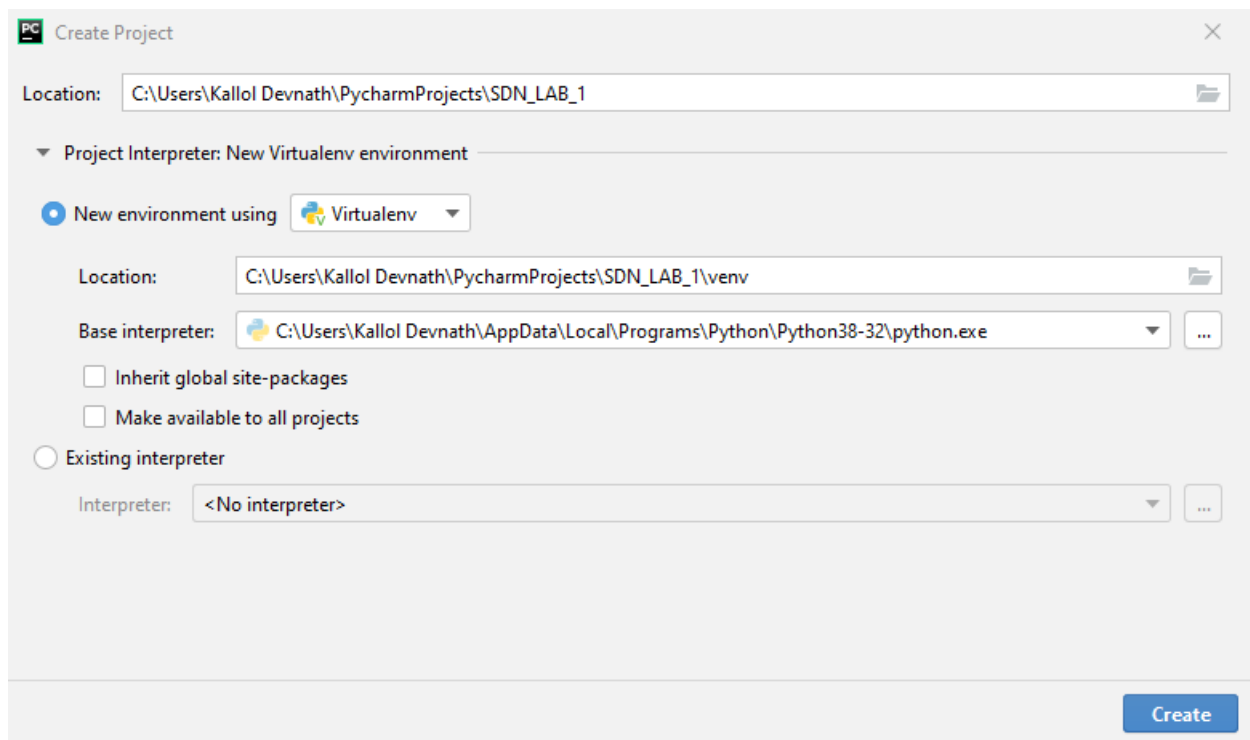


4. Exercises

Section 4.1: Basics of python and programing

Exercise 4.1.1: Create a python project, click in File > New > PyDev Project. Provide a name for the project (SDN_LAB_1 for the fits lab), then select the version of python to be used and select to add the project to working set as shown below:

Ans:



Exercise 4.1.2:

Write a Hello World program Almost all books about programming languages start with a very simple program that prints the text **Hello, World!** to the screen. Make such a program in Python. (save as `hello_world.py`).

Ans:



Exercise 4.1.3:

Compute 1+1 The first exercise concerns some very basic mathematics and programming: assign the result of 1+1 to a variable and print the value of that variable (save as 1plus1.py).

Ans:

```
untitled > 1plus1.py
1 x=1
2 y=1
3 print(x+y)
4 print(x)
5 print(y)

Run: 1plus1 x
"C:\Users\Kallol Devnath\PycharmProjects\untitled\venv\Scripts\python
2
1
1
```

Exercise 4.1.4:

Type in program text

Type the following program in your editor and execute it. If your program does not work, check that you have copied the code correctly and debug it (save as formulas_shapes.py).

Ans:

The screenshot displays the PyCharm IDE interface. The left sidebar shows the project structure with files like `1plus1.py`, `formulas_shapes.py`, and `k.py`. The main editor window shows the code in `formulas_shapes.py`:

```
1 from math import pi
2 h = 5.0 # height
3 r = 1.5 # radius
4 b = 3 # base
5 if __name__ == '__main__':
6     area_parallelogram = h * b
7     print('The area of the parallelogram is %.3f' % area_parallelogram)
8     area_square = b ** 2
9     print('The area of the square is %g' % area_square)
10    area_circle = pi * r ** 2
11    print('The area of the circle is %.3f' % area_circle)
12    volume_cone = 1.0 / 3 * pi * r ** 2 * h
13
14    print('The volume of the cone is %.3f' % volume_cone)
15
```

Below the editor, the 'Run' console shows the execution output for `formulas_shapes`:

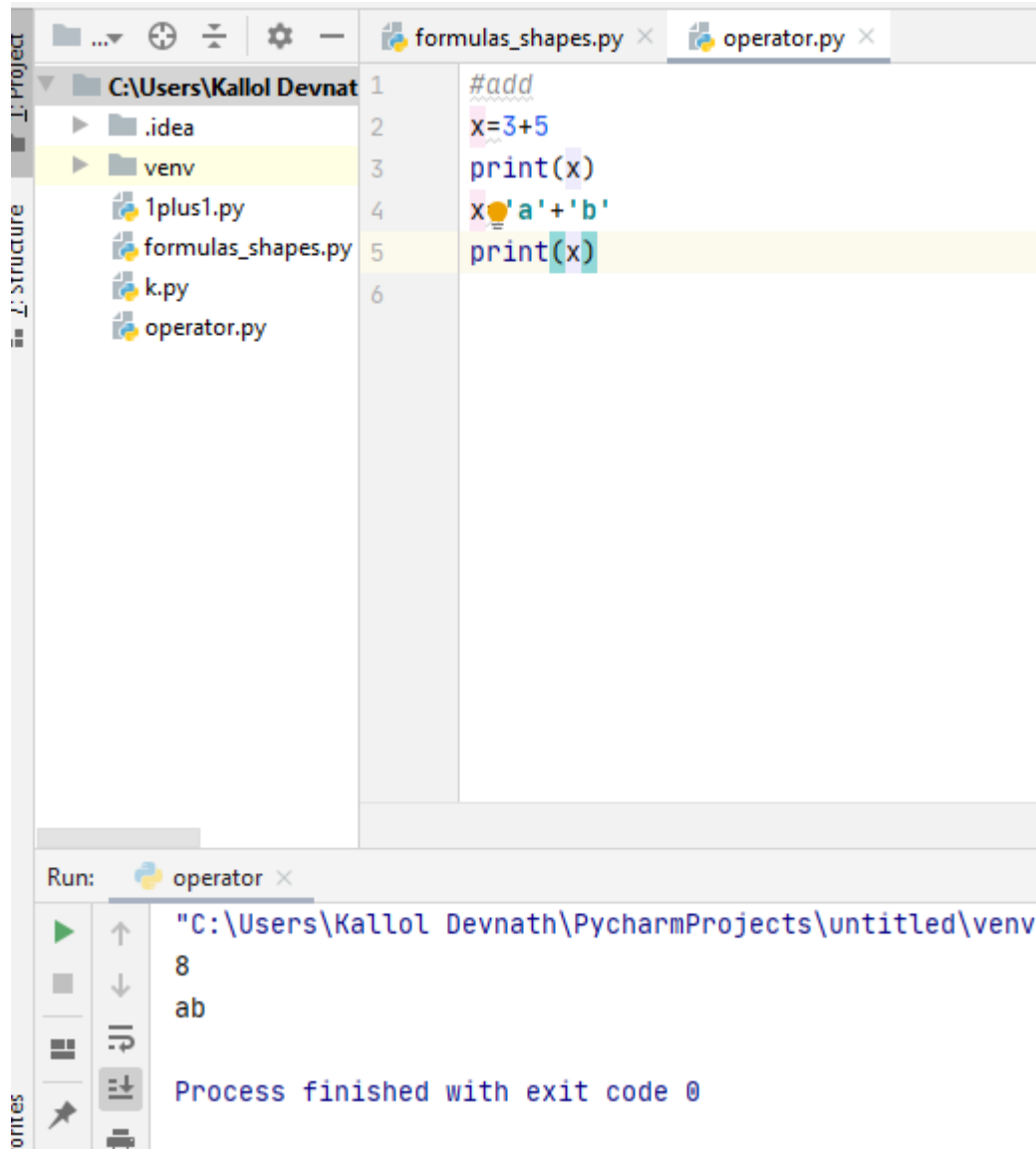
```
"C:\Users\Kallol Devnath\PycharmProjects\untitled\venv\Scripts\python.exe" "C:/Users/Kallol Devna
The area of the parallelogram is 15.000
The area of the square is 9
The area of the circle is 7.069
The volume of the cone is 11.781

Process finished with exit code 0
```

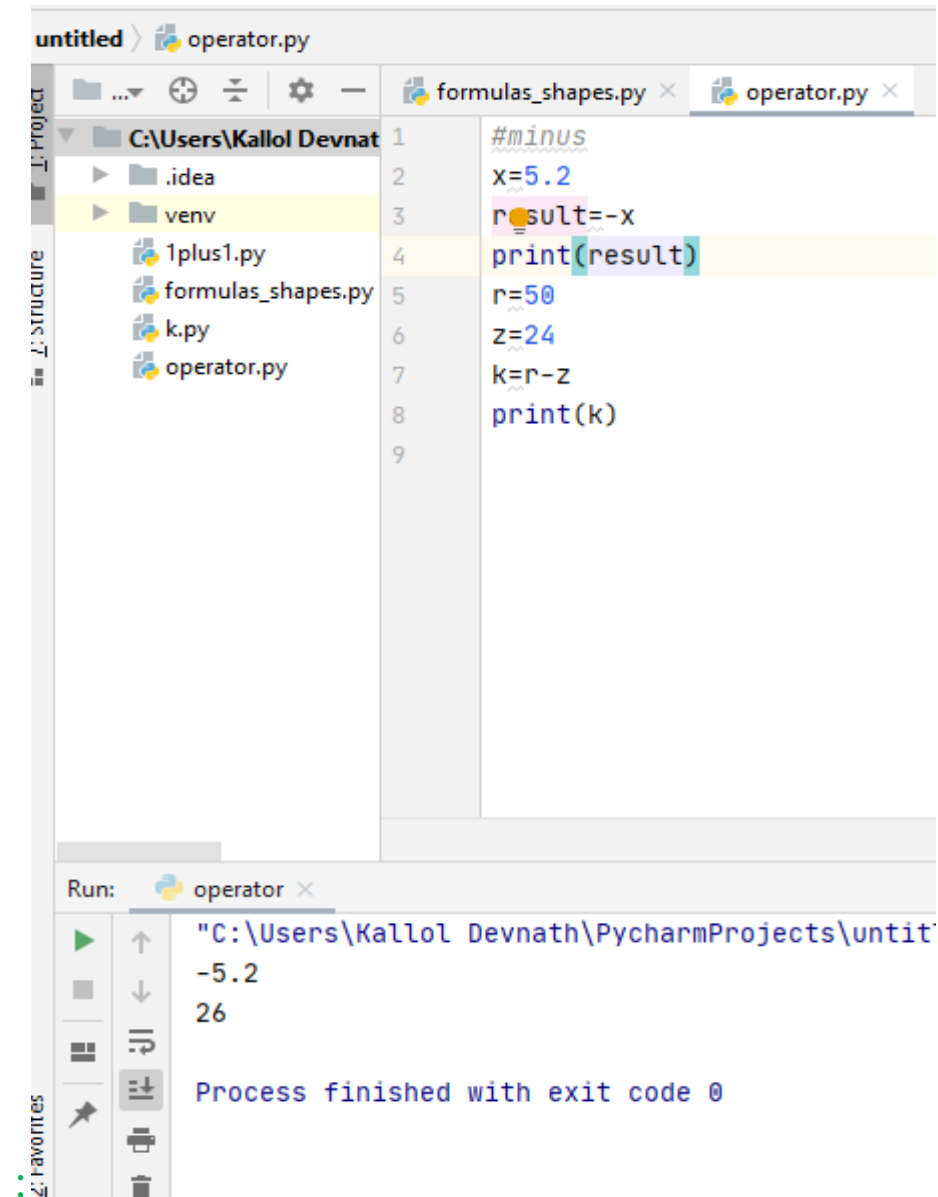
Exercise 4.2.1:

Verify the use of the following operator. Execute the example code in python script and provide the output.

Plus:



Minus:



The screenshot shows the PyCharm IDE interface. The main editor window displays a Python file named `operator.py` with the following code:

```
1 #minus
2 x=5.2
3 result=-x
4 print(result)
5 r=50
6 z=24
7 k=r-z
8 print(k)
9
```

The left sidebar shows the project structure for `C:\Users\Kallol Devnat`, including folders `.idea` and `venv`, and files `1plus1.py`, `formulas_shapes.py`, `k.py`, and `operator.py`.

At the bottom, the Run console shows the output of the script:

```
Run: operator x
"C:\Users\Kallol Devnath\PycharmProjects\untit
-5.2
26
Process finished with exit code 0
```

Power:

The screenshot shows the PyCharm IDE interface. The top toolbar contains icons for file operations (New, Open, Save, Close), settings, and a search icon. The top panel displays two open files: `formulas_shapes.py` and `operator.py`. The left sidebar shows the project structure for `C:\Users\Kallol Devnat`, including folders `.idea` and `venv`, and files `1plus1.py`, `formulas_shapes.py`, `k.py`, and `operator.py`. The main editor area shows the content of `operator.py` with line numbers 1 through 6. The code is as follows:

```
1 #power
2 x=3
3 y=4
4 result=x**y
5 print(result)
6 |
```

The bottom panel shows the Run output for the `operator` process. It displays the command path `"C:\Users\Kallol Devnath\PycharmProjects\untitled"`, the output `81`, and the status `Process finished with exit code 0`.

Exercise 4.2.2:

The if statement:

Ans:

The screenshot shows a Python IDE with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with files like 1plus1.py, formulas_shapes.py, if.py, k.py, and operator.py. The code editor displays the following Python code:

```
1 a = 200
2 b = 33
3 if b > a:
4     print("b is greater than a")
5 elif a == b:
6     print("a and b are equal")
7 else:
8     print("a is greater than b")
```

The output window at the bottom shows the execution of the code, displaying the message "a is greater than b" and indicating that the process finished with exit code 0.

Exercise 4.2.4:

The for Statement Create a program for printing a sequence of numbers. Save the file as for.py

Ans:

