

## Answer Sheet - Python

Name: Isha Garg

Phone No.: 7053017594

Email: ishagarg989@gmail.com

1. Find the datatype of these two declarations:

`x = 5`

`y = "John"`

**Code:**

```
print(type(x))  
print(type(y))
```

**Output:**

The screenshot shows a Jupyter Notebook interface. At the top, it says "jupyter Assignment" and "Last Checkpoint: 6 minutes ago (autosaved)". There is a "Logout" button and a Python logo. Below this is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar are "Trusted" and "Python 3 (ipykernel)". Below the menu bar is a toolbar with icons for saving, adding cells, undo, redo, and running code. The main area shows two input cells. The first cell, labeled "In [2]:", contains the code `x = 5` and `print(type(x))`, and its output is `<class 'int'>`. The second cell, labeled "In [3]:", contains the code `y = "John"` and `print(type(y))`, and its output is `<class 'str'>`.

# X is int because it is integer number without decimal.

# Y is str because it is enclosed in double quotes (“”).

2. Check whether the following syntax is valid or invalid for naming a variable:

Example: `abc = 100` #valid syntax

- i. `3a=10`
- ii. `@abc=10`
- iii. `a100=100`
- iv. `_a984_=100`
- v. `a9967$=100`
- vi. `xyz-2=100`

## Code:

**`3a=10`**

# Invalid syntax (because variable name cannot start with number)

**`@abc=10`**

# Invalid syntax (because variable name cannot start with special character except underscore (\_))

**`a100=100`**

# Valid syntax (because variable name must start with alphabet or underscore (\_))

**`_a984_=100`**

# Valid syntax (because variable name must start with alphabet or underscore (\_))


**`a9967$=100`**

# Invalid syntax (because variable name cannot have special character except underscore (\_))


**`xyz-2=100`**

# Invalid syntax (because variable name cannot have special character except underscore (\_))











# Output:

 Assignment


Last Checkpoint: 17 minutes ago (unsaved changes)

 Logout

FileEditViewInsertCellKernelWidgetsHelpTrustedPython 3 (ipykernel)



Code



In [5]:

```
3a=10
```

Cell In[5], line 1  
3a=10  
^  
SyntaxError: invalid decimal literal

In [6]:

```
@abc=10
```

Cell In[6], line 1  
@abc=10  
^  
SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?

In [7]:

```
a100=100
```

In [8]:

```
_a984_=100
```

Activate Windows  
Go to Settings to activate Windows.

In [9]:

```
a9967$=100
```

Cell In[9], line 1  
a9967\$=100  
^  
SyntaxError: invalid syntax

In [10]:

```
xyz-2=100
```

Cell In[10], line 1  
xyz-2=100  
^  
SyntaxError: cannot assign to expression here. Maybe you meant '==' instead of '='?

### 3. Check if element exists in list in Python:

```
list = test_list = [1,6,3,5,3,4]
```

1) Check if 3 exist or not.



2) Check if 9 exists or not

#### Code:













```
print(3 in list)
```

```
print(9 in list)
```

#### Output

 jupyter Assignment Last Checkpoint: 36 minutes ago (autosaved)  Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

        Run    Code 

```
In [19]: list = test_list = [1,6,3,5,3,4]
         print(3 in list)
```

```
True
```

```
In [20]: print(9 in list)
```

```
False
```

3) Take the user input to print the current date

## Code:

```
from datetime import date

user = input('If you want to know the current date, Press Y: ')

if user == 'Y' or user == 'y':
    current_date = date.today()
    print(f'Current Date: {current_date}')_date}')
```

## Output:



The image shows a Jupyter Notebook interface. At the top, it says "Jupyter Python Assign code" and "Last Checkpoint: a minute ago (unsaved changes)". There is a "Logout" button. Below the header is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar is a "Not Trusted" warning and "Python 3 (ipykernel)". Below the menu bar is a toolbar with icons for file operations, running, and other notebook functions. The main area shows a code cell with the following code:

```
In [11]: from datetime import date
user = input('If you want to know the current date, Press Y: ')
if user == 'Y' or user == 'y':
    current_date = date.today()
    print(f'Current Date: {current_date}')
```

Below the code cell, the output is displayed:

```
If you want to know the current date, Press Y: Y
Current Date: 2023-06-28
```

4. What is the output of the following codes?

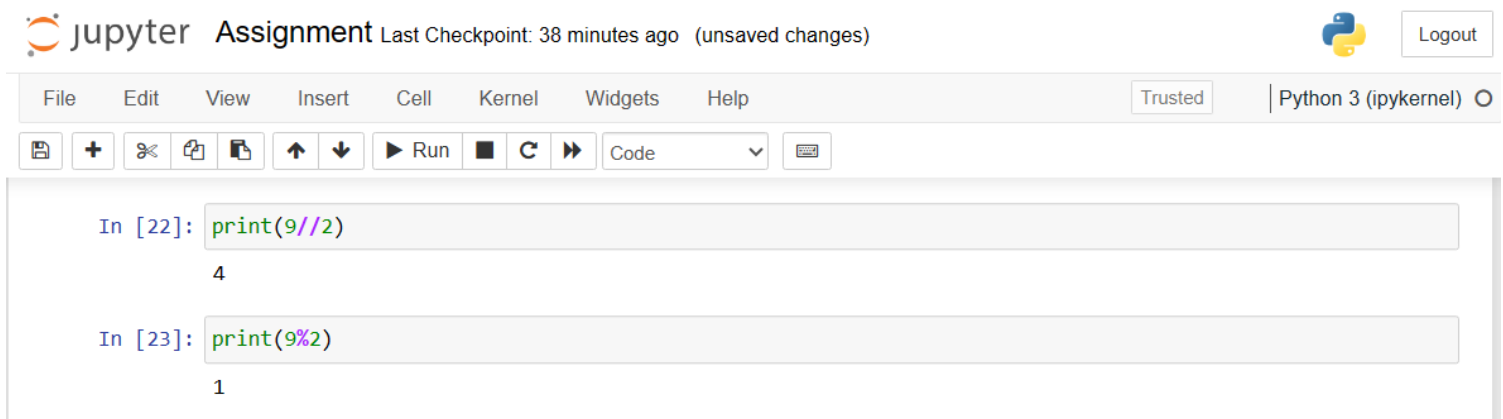
a. `print 9//2`

b. `print 9%2`

```
print(9//2)
```

```
print(9%2)
```

## Output:



The image shows a Jupyter Notebook interface. At the top, the header includes the Jupyter logo, the word "Assignment", and a status message "Last Checkpoint: 38 minutes ago (unsaved changes)". On the right, there is a "Logout" button and a Python logo. Below the header is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. To the right of the menu bar are buttons for "Trusted" and "Python 3 (ipykernel)". Below the menu bar is a toolbar with icons for saving, adding a new cell, undo, redo, copy, paste, and other standard Jupyter actions. The main area of the notebook contains two code cells. The first cell, labeled "In [22]:", contains the code `print(9//2)` and its output is `4`. The second cell, labeled "In [23]:", contains the code `print(9%2)` and its output is `1`.

jupyter Assignment Last Checkpoint: 38 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [22]: `print(9//2)`  
4

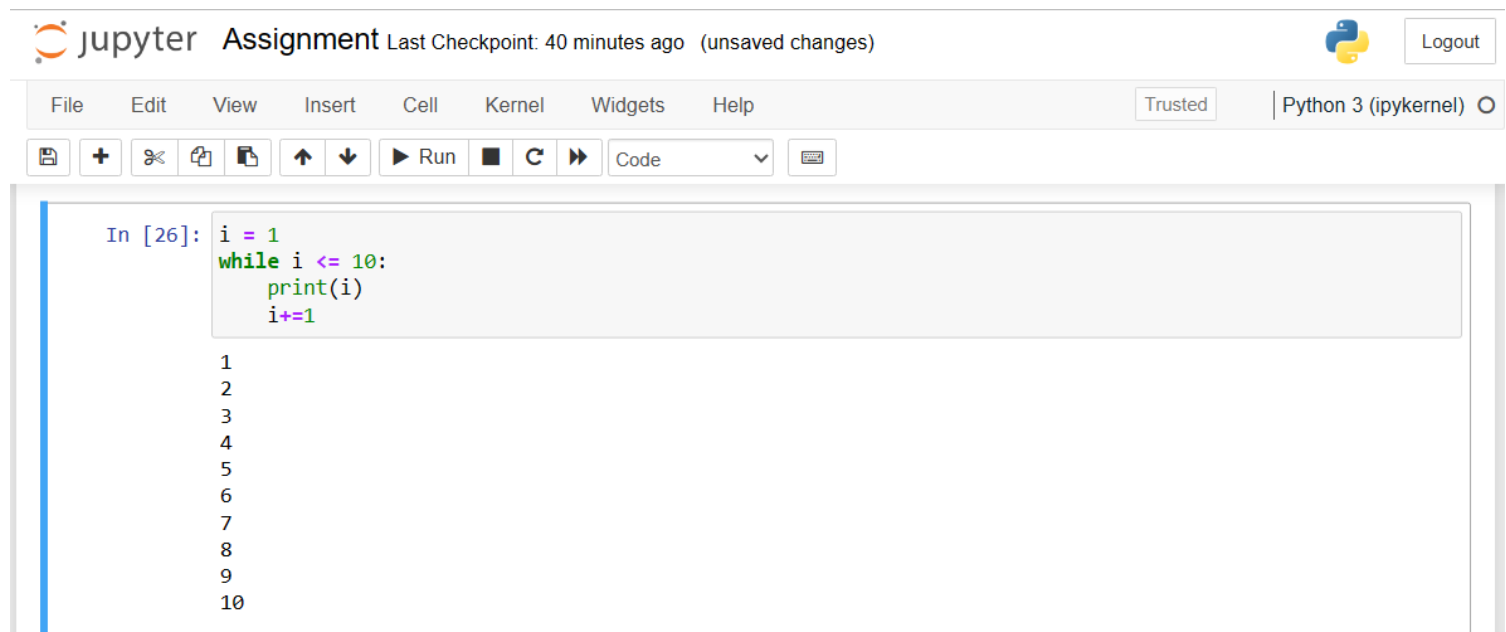
In [23]: `print(9%2)`  
1

## 5. Print First 10 natural numbers using a while loop

### Code:

```
i = 1
while i <= 10:
    print(i)
    i+=1
```

### Output:



The image shows a Jupyter Notebook interface. At the top, the header includes the Jupyter logo, the word "Assignment", and the text "Last Checkpoint: 40 minutes ago (unsaved changes)". On the right, there is a "Logout" button. Below the header is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. To the right of the menu bar are buttons for "Trusted" and "Python 3 (ipykernel)". Below the menu bar is a toolbar with icons for saving, adding cells, undo, redo, and other functions. The main area of the notebook shows a code cell with the following code:

```
In [26]: i = 1
while i <= 10:
    print(i)
    i+=1
```

The output of the code is displayed below the code cell, showing the numbers 1 through 10, each on a new line.

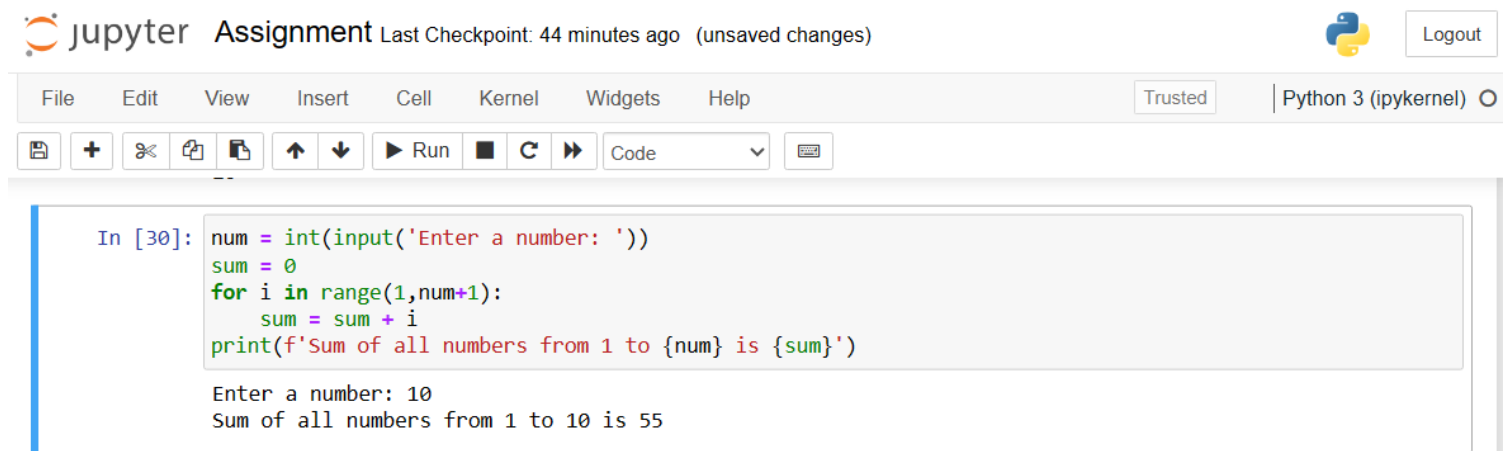
6. Write a program to accept a number from a user and calculate the sum of all numbers from 1 to a given number.

For example, if the user entered 10 the output should be 55(1+2+3+4+5+6+7+8+9+10).

### Code:

```
num = int(input('Enter a number: '))
sum = 0
for i in range(1,num+1):
    sum = sum + i
print(f'Sum of all numbers from 1 to {num} is {sum}')
```

### Output:



The image shows a Jupyter Notebook interface. At the top, the header includes the Jupyter logo, the word "Assignment", and "Last Checkpoint: 44 minutes ago (unsaved changes)". On the right, there is a "Logout" button. Below the header is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar are "Trusted" and "Python 3 (ipykernel)". Below the menu bar is a toolbar with icons for saving, adding cells, undo, redo, running, and other functions. The main area shows a code cell with the following code:

```
In [30]: num = int(input('Enter a number: '))
sum = 0
for i in range(1,num+1):
    sum = sum + i
print(f'Sum of all numbers from 1 to {num} is {sum}')
```

The output of the code is displayed below the code cell:

```
Enter a number: 10
Sum of all numbers from 1 to 10 is 55
```



7. Write a Python program which iterates the integers from 1 to 50. For multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

Example:

Fizzbuzz

1

2

Fizz

4

buzz

### Code:

```
for i in range(0,51):
    if i%3==0 and i%5==0:
        print('FizzBuzz')
    elif i%3==0:
        print('Fizz')
    elif i%5==0:
        print('Buzz')
    else:
        print(i)
```

# Output:

 jupyter

Untitled

Last Checkpoint: 3 minutes ago (unsaved changes)

 Logout

FileEditViewInsertCellKernelWidgetsHelpTrustedPython 3 (ipykernel)



Code



```
In [2]: for i in range(0,51):
        if i%3==0 and i%5==0:
            print('FizzBuzz')
        elif i%3==0:
            print('Fizz')
        elif i%5==0:
            print('Buzz')
        else:
            print(i)
```

FizzBuzz  
1  
2  
Fizz  
4  
Buzz  
Fizz  
7  
8  
Fizz  
Buzz  
11

Fizz  
13  
14  
FizzBuzz  
16  
17  
Fizz  
19  
Buzz  
Fizz  
22  
23  
Fizz  
Buzz  
26  
Fizz  
28  
29  
FizzBuzz  
31  
32  
Fizz  
34

Buzz  
Fizz  
37  
38  
Fizz  
Buzz  
41  
Fizz  
43  
44  
FizzBuzz  
46  
47  
Fizz  
49  
Buzz

Activate Windows  
Go to Settings to activate Windows

Activate Windows  
Go to Settings to activate Windows