

Task-8

Question-1:

The screenshot shows a Replit Python environment. The left sidebar contains a 'Files' panel with 'main.py' and 'Packager files' (poetry.lock, pyproject.toml). The main editor displays the following Python code in 'main.py':

```
1 import numpy as np
2
3 vector = np.array([10, 11, 12, 13, 14])
4 print("Given vector")
5 print(vector)
6 p = 5
7 new_vector = np.zeros(len(vector) + (len(vector)-1)*(p))
8 new_vector[::p+1] = vector
9 print("\nNew vector:")
10 print(new_vector)
```

The right panel shows the 'Console' output:

```
Given vector
[10 11 12 13 14]

New vector:
[10.  0.  0.  0.  0.  0. 11.  0.  0.  0.  0.  0. 12.  0.  0.  0.  0.
 13.  0.  0.  0.  0.  0. 14.]
> []
```

At the bottom, there are status indicators for CPU, RAM, and Storage.

Question-2:

The screenshot shows a Replit web interface. At the top, the browser address bar displays 'replit.com/@SaiYeswanth1/q2#main.py'. The interface includes a top navigation bar with a 'Run' button and an 'Invite' button. On the left, a 'Files' sidebar shows a project structure with 'main.py' and 'Packager files' (poetry.lock, pyproject.toml). The main editor area displays a Python script in 'main.py' with the following code:

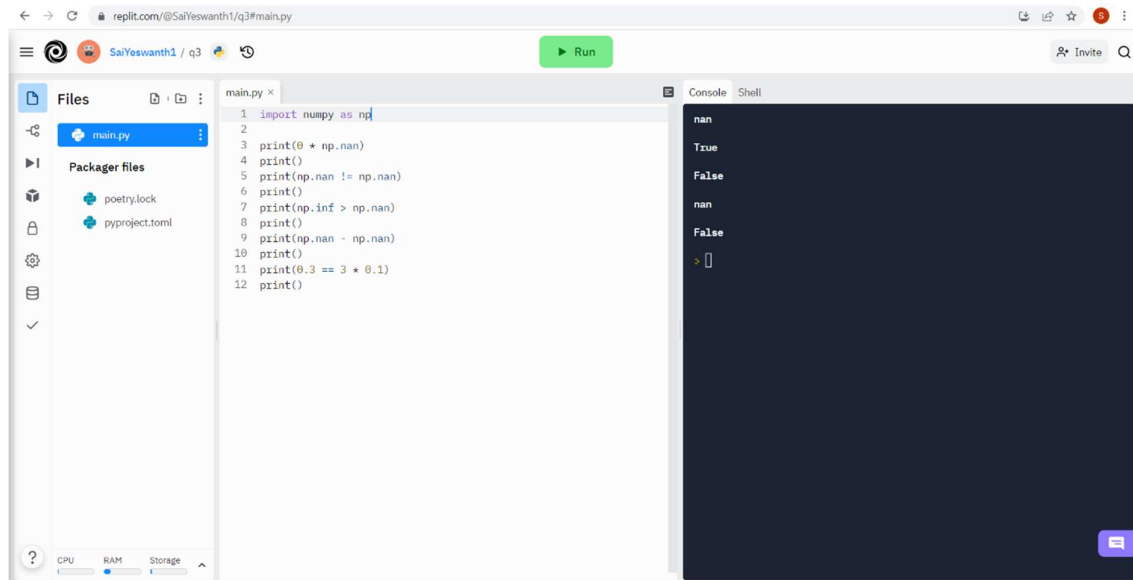
```
1 import numpy as np
2
3 x = input(np.array)
4 print('First array:')
5 print(x)
6
7 y = input(np.array)
8 print('Second array:')
9 print(y)
10
11 print('Testing if the above two arrays are equal or not!')
12 array_equal = np.array_equal(np.array(x), np.array(y))
13
14 print(array_equal)
```

On the right, the 'Console' tab shows the output of the script:

```
<built-in function array>[1 0 0 0 1 0]
First array:
[1 0 0 0 1 0]
<built-in function array>[0 0 1 1 0 1]
Second array:
[0 0 1 1 0 1]
Testing if the above two arrays are equal or not!
False
>
```

At the bottom left, a status bar shows system metrics: CPU, RAM, and Storage.

Question-3:



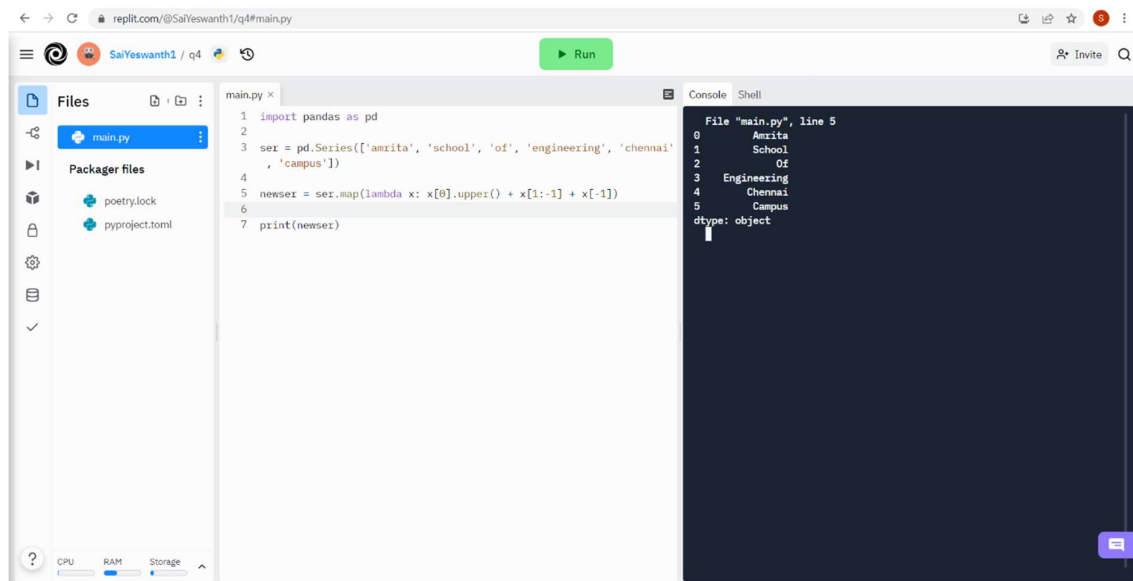
The screenshot shows a Replit environment for a Python script named `main.py`. The script uses `numpy` to perform various operations on `np.nan`. The console output shows the results of these operations.

```
1 import numpy as np
2
3 print(0 + np.nan)
4 print()
5 print(np.nan != np.nan)
6 print()
7 print(np.inf > np.nan)
8 print()
9 print(np.nan - np.nan)
10 print()
11 print(0.3 == 3 * 0.1)
12 print()
```

Console Output:

```
nan
True
False
nan
False
> []
```

Question-4:



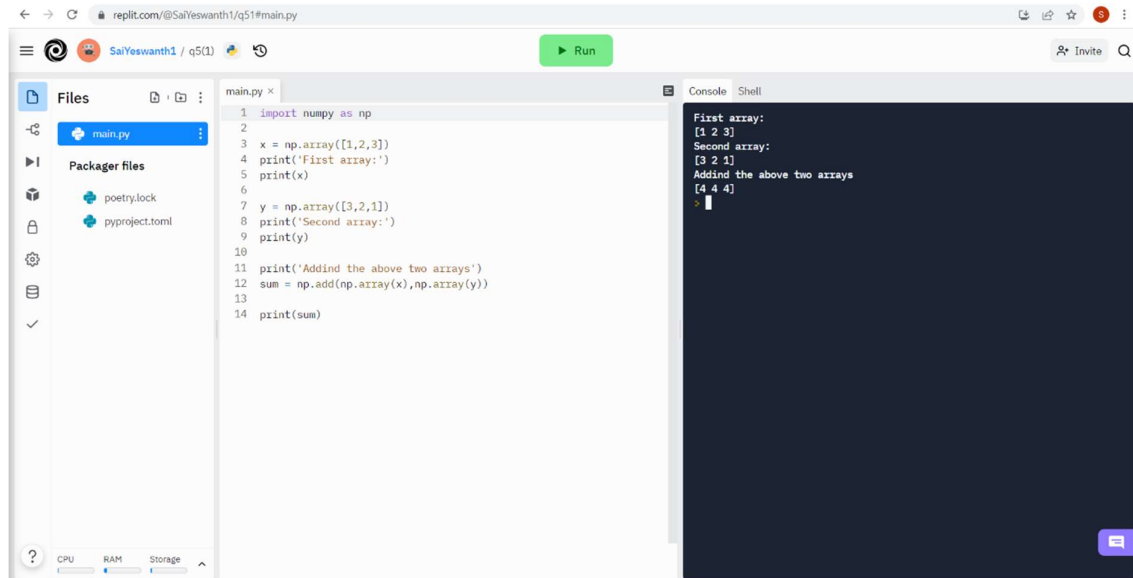
The screenshot shows a Replit environment for a Python script named `main.py`. The script uses `pandas` to create a series and apply a lambda function. The console output shows the resulting series and its dtype.

```
1 import pandas as pd
2
3 ser = pd.Series(['amrita', 'school', 'of', 'engineering', 'chennai',
4 , 'campus'])
5 newser = ser.map(lambda x: x[0].upper() + x[1:-1] + x[-1])
6
7 print(newser)
```

Console Output:

```
File "main.py", line 5
0      Amrita
1      School
2      Of
3      Engineering
4      Chennai
5      Campus
dtype: object
```

Question-5(1):

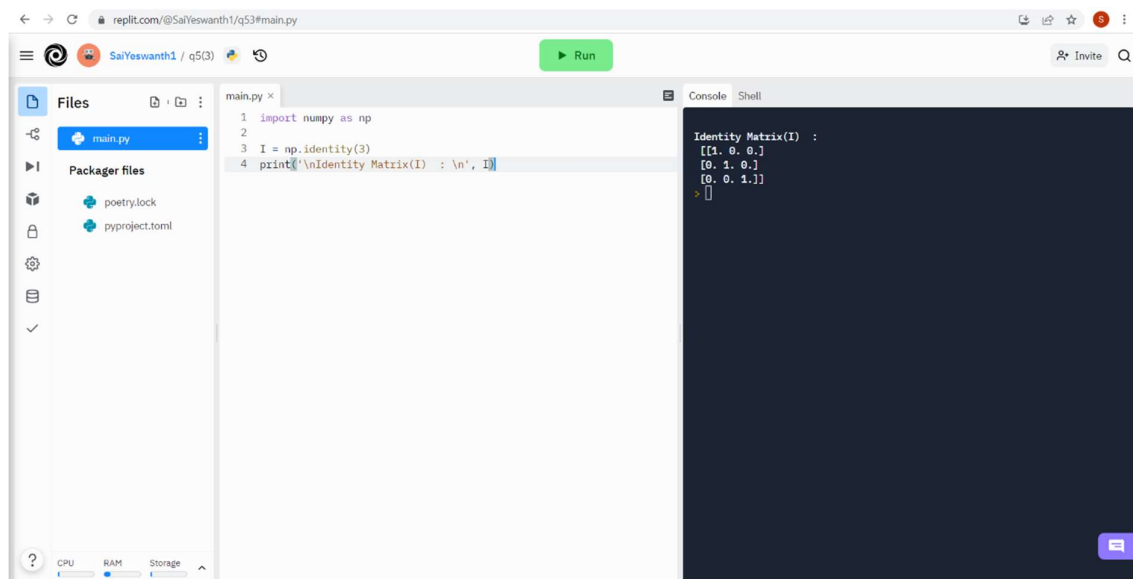


The screenshot shows a Replit environment with a Python script in `main.py` and its output in the console. The script imports `numpy` as `np`, creates two arrays `x` and `y`, prints them, adds them, and prints the result.

```
1 import numpy as np
2
3 x = np.array([1,2,3])
4 print('First array:')
5 print(x)
6
7 y = np.array([3,2,1])
8 print('Second array:')
9 print(y)
10
11 print('Adding the above two arrays')
12 sum = np.add(np.array(x), np.array(y))
13
14 print(sum)
```

The console output shows the first array `[1 2 3]`, the second array `[3 2 1]`, and the resulting sum `[4 4 4]`.

Question-5(3):



The screenshot shows a Replit environment with a Python script in `main.py` and its output in the console. The script imports `numpy` as `np`, creates a 3x3 identity matrix `I`, and prints it.

```
1 import numpy as np
2
3 I = np.identity(3)
4 print('\nIdentity Matrix(I) : \n', I)
```

The console output shows the identity matrix `I` as a 3x3 array of ones and zeros.