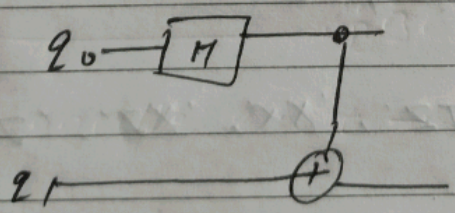
DAY-14 TASK.

Coding with qiskit.

1. Map the Problem to circuits and operators
2. optimize the circuit
3. Execute it on a backend
4. Post-Process the Result

Step1:- Map the Problem to circuits and operators

```
from qiskit import QuantumCircuit
qc = QuantumCircuit(2)
qc.h(0)
qc.cx(0,1)
qc.draw(output='mpl')
```




```
from qiskit.quantum_info import Pauli
ZZ = Pauli('ZZ')
ZI = Pauli('ZI')
```

```
IZ = Pauli('IZ')
XX = Pauli('XX')
XI = Pauli('XI')
IX = Pauli('IX')

observables = (ZZ, ZI, IZ, XX, XI, IX)
```

Step 2: optimize

Step 3: Execute on the backend

```
from qiskit_ibm_runtime import Estimator

estimator = Estimator()

job = estimator.run([qc] * len(observables),
                    observables)

job.result()
```

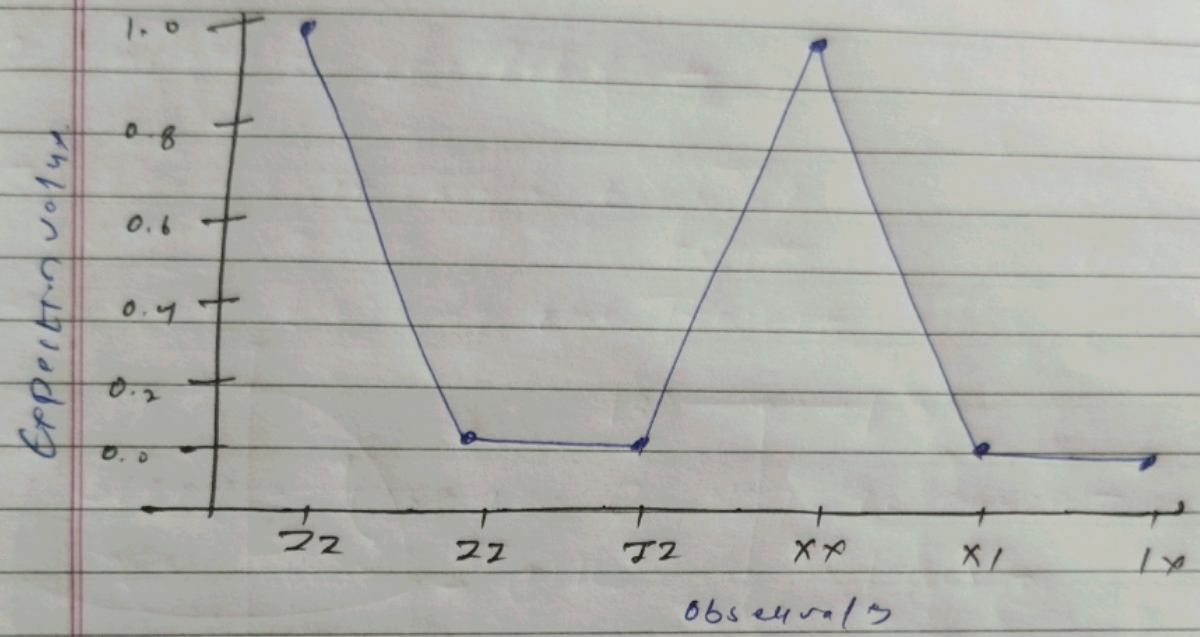
Step 4: Post-Process

```
import matplotlib.pyplot as plt

data = ['ZZ', 'ZI', 'IZ', 'XX', 'XI', 'IX']

values = job.result().values

plt.plot(data, values, '-o')
```

Date
Page No.

plt.xlabel('observables')
plt.ylabel('expectation value')

plt.show()



Primitive:
The smallest processing instruction for a given abstraction level.

Sampler Primitive:

Returns shot by shot bit strings sampled from the probability distribution of the quantum state prepared on the device