In [4]:

```
from qiskit import QuantumRegister, ClassicalRegister
from qiskit import QuantumCircuit, execute,IBMQ,Aer
from qiskit.tools.monitor import job_monitor
from qiskit.circuit.library import QFT
import numpy as np

pi = np.pi

# Loading your IBM Quantum account(s)
provider = IBMQ.load_account()
```

```
ibmqfactory.load_account:WARNING:2021-07-19 13:01:00,898: Credentials are already in use. The exi
sting account in the session will be replaced.
```

In [5]:

```
q = QuantumRegister(5,'q')
c = ClassicalRegister(5,'c')

circuit = QuantumCircuit(q,c)

circuit.x(q[4])
circuit.x(q[2])
circuit.x(q[0])
circuit += QFT(num_qubits=5, approximation_degree=0, do_swaps=True, inverse=False, insert_barriers=
False, name='qft')
circuit.measure(q,c)
circuit.draw(output='mpl', filename='qft1.png')
print(circuit)
```

```
     ┌───┐                                                                 »
q_0: ┤ X ├───────────────────────────────────────────────■─────────────────»
     └───┘                                               │                 »
q_1: ──────────────────────────────────■─────────────────┼─────────■───────»
     ┌───┐                             │                 │         │       »
q_2: ┤ X ├───────────────■─────────────┼────────■────────┼─────────┼───────»
     └───┘               │       ┌───┐ │        │P(π/2)  │         │P(π/4) »
q_3: ───────────■────────┼───────┤ H ├─┼────────■────────┼─────────■───────»
     ┌───┐┌───┐ │P(π/2)  │P(π/4) └───┘ │P(π/8)           │P(π/16)          »
q_4: ┤ X ├┤ H ├─■────────■─────────────■─────────────────■─────────────────»
     └───┘└───┘                                                            »
c: 5/══════════════════════════════════════════════════════════════════════»
                                                                           »
«                                                   ┌───┐         ┌─┐    
«q_0: ──────■─────────────────■─────────────■───────┤ H ├─X───────┤M├────
«           │                 │       ┌───┐ │P(π/2) └───┘ │ ┌─┐   └╥┘    
«q_1: ──────┼────────■────────┼───────┤ H ├─■─────────X───┼─┤M├────╫─────
«     ┌───┐ │        │P(π/2)  │P(π/4) └┬─┬┘           │   │ └╥┘    ║     
«q_2: ┤ H ├─┼────────■────────■────────┤M├────────────┼───┼──╫─────╫─────
«     └───┘ │P(π/8)                    └╥┘            │   │  ║ ┌─┐ ║     
«q_3: ──────■───────────────────────────╫─────────────X───┼──╫─┤M├─╫─────
«                                       ║                 │  ║ └╥┘ ║ ┌─┐ 
«q_4: ──────────────────────────────────╫─────────────────X──╫──╫──╫─┤M├─
«                                       ║                    ║  ║  ║ └╥┘ 
«c: 5/══════════════════════════════════╩════════════════════╩══╩══╩══╩══
«                                       2                    1  3  0  4
```

In [6]:

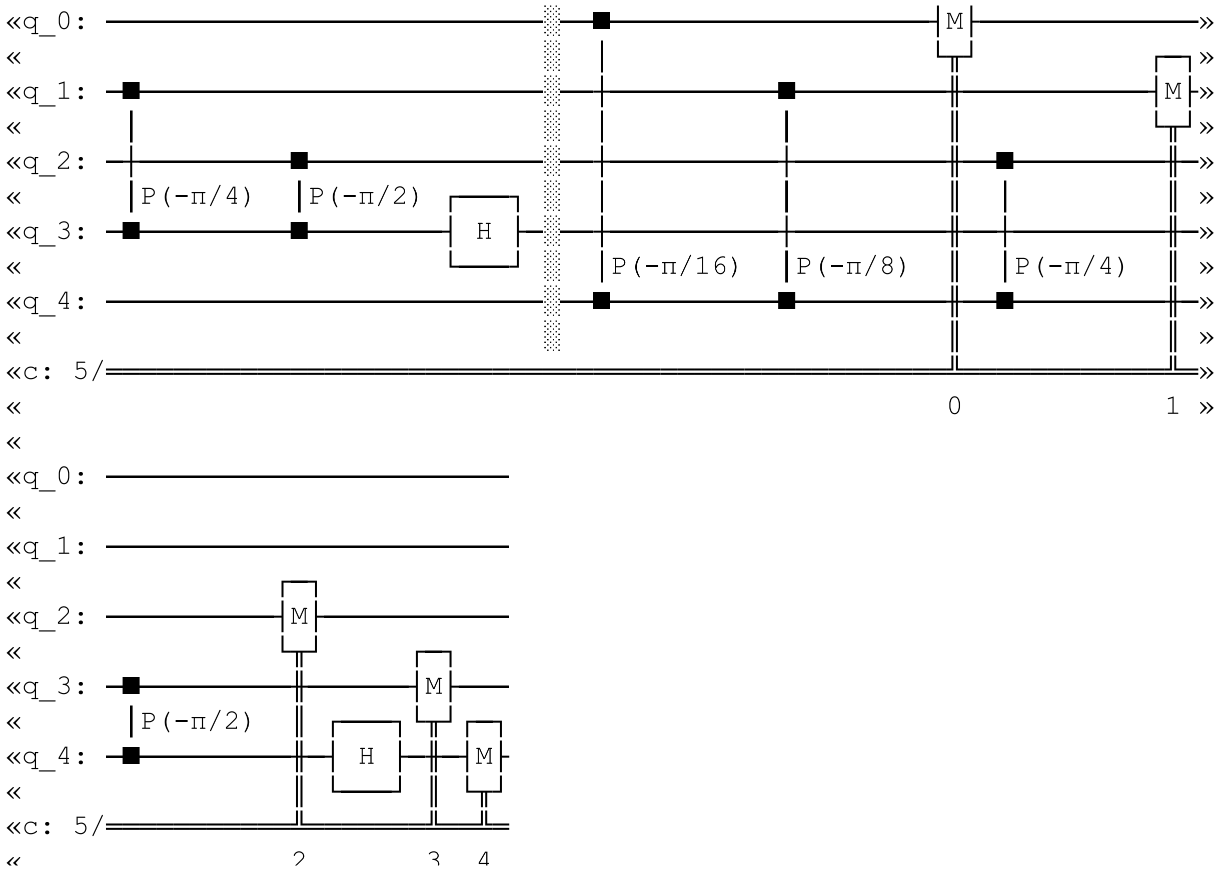
```
backend = Aer.get_backend('aer_simulator')

job = execute(circuit, backend, shots=1000)

job_monitor(job)

counts = job.result().get_counts()

print("\n QFT Output")
print("-------------")
print(counts)
```

```
Job Status: job has successfully run

 QFT Output
-------------
{'10011': 21, '11011': 32, '00110': 23, '00010': 43, '01100': 37, '01000': 38, '01111': 43, '0001
1': 30, '00100': 27, '00111': 26, '11010': 37, '11000': 22, '10111': 24, '11111': 45, '10010': 3
5, '10110': 36, '11110': 30, '11100': 38, '10000': 30, '10101': 27, '01110': 23, '01101': 29, '00
001': 23, '01001': 33, '00101': 36, '00000': 30, '01011': 38, '10100': 23, '11101': 34, '11001':
30, '10001': 26, '01010': 31}
```

In [8]:

```
#input()

q = QuantumRegister(5,'q')
c = ClassicalRegister(5,'c')

circuit = QuantumCircuit(q,c)

circuit.x(q[4])
circuit.x(q[2])
circuit.x(q[0])
circuit += QFT(num_qubits=5, approximation_degree=0, do_swaps=True, inverse=False, insert_barriers=True, name='qft')
circuit += QFT(num_qubits=5, approximation_degree=0, do_swaps=True, inverse=True, insert_barriers=True, name='qft')
circuit.measure(q,c)
circuit.draw(output='mpl',filename='qft2.png')

print(circuit)
```

```
     ┌───┐                                           ░                        »
q_0: ┤ X ├─────────────────────────────────■─────────░────────────────────────»
     └───┘                                 │         ░                        »
q_1: ─────────────────────────────■────────┼─────────░────────────────■───────»
     ┌───┐                        │        │         ░                │       »
q_2: ┤ X ├───────────────■────────┼────────┼─────────░───────■────────┼───────»
     └───┘               │        │        │         ░ ┌───┐ │P(π/2)  │P(π/4) »
q_3: ───────────■────────┼────────┼────────┼─────────░─┤ H ├─■────────■───────»
     ┌───┐┌───┐ │P(π/2)  │P(π/4)  │P(π/8)  │P(π/16)  ░ └───┘                  »
q_4: ┤ X ├┤ H ├─■────────■────────■────────■─────────░────────────────────────»
     └───┘└───┘                                      ░                        »
c: 5/═════════════════════════════════════════════════════════════════════════»
                                                                              »
«               ░                         ░                ░ ┌───┐ ░          »
«q_0: ─■────────░────────────────■────────░───────■────────░─┤ H ├─░──X───────»
«      │        ░                │        ░ ┌───┐ │P(π/2)  ░ └───┘ ░  │       »
«q_1: ─┼────────░───────■────────┼────────░─┤ H ├─■────────░───────░──┼──X──X─»
«      │        ░ ┌───┐ │P(π/2)  │P(π/4)  ░ └───┘          ░       ░  │  │  │ »
«q_2: ─┼────────░─┤ H ├─■────────■────────░────────────────░───────░──┼──┼──┼─»
«      │P(π/8)  ░ └───┘                   ░                ░       ░  │  │  │ »
«q_3: ─■────────░─────────────────────────░────────────────░───────░──┼──X──X─»
«               ░                         ░                ░       ░  │       »
«q_4: ──────────░─────────────────────────░────────────────░───────░──X───────»
«               ░                         ░                ░       ░          »
«c: 5/════════════════════════════════════════════════════════════════════════»
«                                                                             »
«         ░ ┌───┐ ░                 ░                           ░           »
«q_0: ─X──░─┤ H ├─░──■──────────────░──■────────────────────────░──■────────»
«      │  ░ └───┘ ░  │P(-π/2) ┌───┐ ░  │                        ░  │        »
«q_1: ─┼──░───────░──■────────┤ H ├─░──┼─────────■──────────────░──┼────────»
«      │  ░       ░           └───┘ ░  │P(-π/4)  │P(-π/2) ┌───┐ ░  │        »
«q_2: ─┼──░───────░─────────────────░──■─────────■────────┤ H ├─░──┼────────»
«      │  ░       ░                 ░                     └───┘ ░  │P(-π/8) »
«q_3: ─┼──░───────░─────────────────░───────────────────────────░──■────────»
«      │  ░       ░                 ░                           ░           »
«q_4: ─X──░───────░─────────────────░───────────────────────────░───────────»
«         ░       ░                 ░                           ░           »
«c: 5/══════════════════════════════════════════════════════════════════════»
«                                                                           »
«                               ░                       ┌─┐            »
```



In [9]:

```
job = execute(circuit, backend, shots=1000)

job_monitor(job)

counts = job.result().get_counts()

print("\n QFT with inverse QFT Output")
print("------------------------------")
print(counts)
```

```
Job Status: job has successfully run

 QFT with inverse QFT Output
------------------------------
{'10101': 1000}
```

In [ ]:

```
This code is a part of Qiskit

© Copyright IBM 2017, 2020.

This code is licensed under the Apache License, Version 2.0. You may obtain a copy of this license
in the LICENSE.txt file in the root directory of this source tree or at http://www.apache.org/licen
ses/LICENSE-2.0.

Any modifications or derivative works of this code must retain this copyright notice, and modified
files need to carry a notice indicating that they have been altered from the originals.

#Program executed by Bhadale IT in IBM Quantum Lab (https://www.bhadaleit.com).
#For more details on the Qiskit code and tutorials visit https://qiskit.org/ website
```