In [1]:

```
import numpy as np

# Importing standard Qiskit libraries
from qiskit import QuantumCircuit, transpile, Aer, IBMQ
from qiskit.tools.jupyter import *
from qiskit.visualization import *
from ibm_quantum_widgets import *
from qiskit.providers.aer import QasmSimulator

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

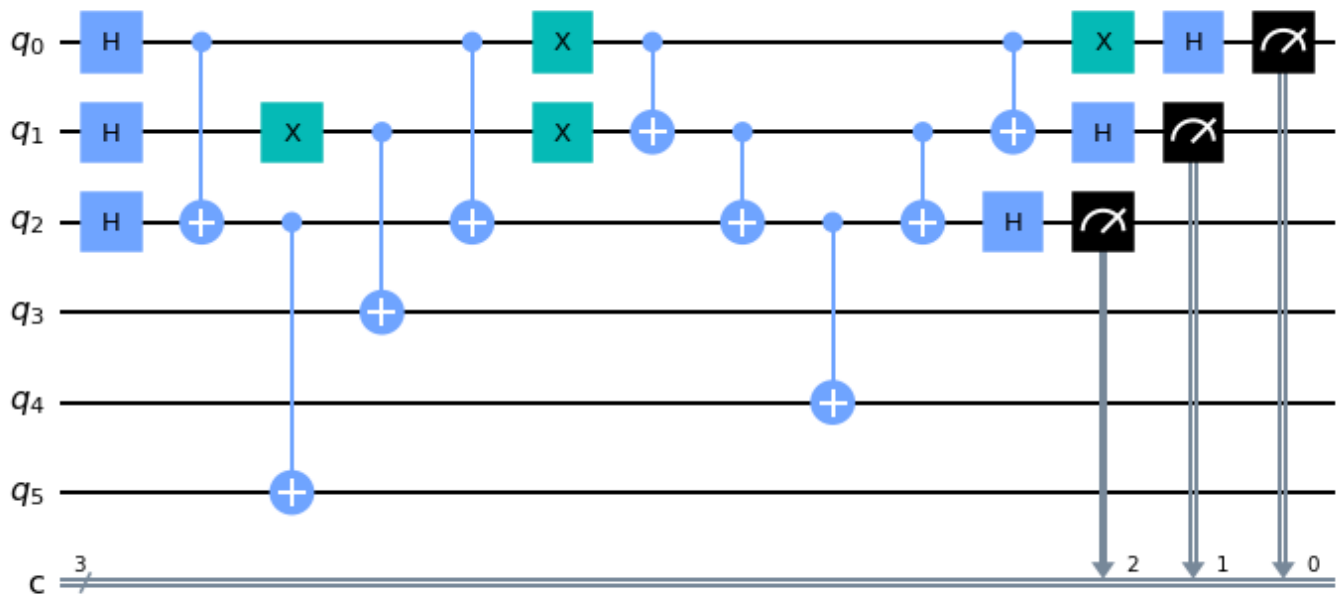
In [9]:

```
from qiskit import*
#building an oracle for the function f(x1,x2,x3)=(1+x2,1+x1+x2+x3,x1+x3)
qr=QuantumRegister(6,'q')
cr=ClassicalRegister(3,'c')
orc=QuantumCircuit(qr,cr)
for i in range(3):
    orc.h(qr[i])
orc.cx(qr[0],qr[2])
orc.x(qr[1])
orc.cx(qr[2],qr[5])
orc.cx(qr[1],qr[3])
orc.cx(qr[0],qr[2])
orc.x(qr[0])
orc.x(qr[1])
orc.cx(qr[0],qr[1])
orc.cx(qr[1],qr[2])
orc.cx(qr[2],qr[4])
orc.cx(qr[1],qr[2])
orc.cx(qr[0],qr[1])
orc.x(qr[0])

for i in range(3):
    orc.h(qr[i])
    orc.measure(qr[i],cr[i])
# Executing the code in the simulator
backend = Aer.get_backend('qasm_simulator')
qjob = execute(orc,backend,shots=3)
counts = qjob.result().get_counts()
print(counts)
orc.draw()
```

```
{'111': 1, '010': 1, '101': 1}
```

Out[9]:



In [ ]:

```
#From the output we use the strings y1=111,y2=010,y3=101.
#To solve the system of linear equations given by yi.s=0, Which implies s=101.
# We can verify that f(000)=f(000+101).
```

In [ ]:

```
# The hidden shift for the given 3-input 3-output function is s=101
```

In [ ]:

```
#Submitted by Tufan Singha Mahapatra(CRS2014)
```