Cedric Evans
9/9/2021
PHYS 306-HW1

Below is the code of the homework for the first six problems. Below all the code are the results.

```
#====  2-2, #6 ================================
print('Problem 2-2, #6')
def BinaryConversion(n):
    return int(n,2)

if __name__ == '__main__':
    print(BinaryConversion('100001'))
    print(BinaryConversion('100111'))
    print(BinaryConversion('101010'))
    print(BinaryConversion('111001'))
    print(BinaryConversion('1100000'))
    print(BinaryConversion('11111101'))
    print(BinaryConversion('11110010'))
    print(BinaryConversion('11111111'))
    #print(BinaryConversion('15'))
print(20*'-')
#============================================
#====== 2-2, #8 ================================
print('Problem 2-2, #8')
def MaxDecimalNumber(n):
    return (2**n)-1

print(MaxDecimalNumber(2))
print(MaxDecimalNumber(3))
print(MaxDecimalNumber(4))
print(MaxDecimalNumber(5))
print(MaxDecimalNumber(6))
print(MaxDecimalNumber(7))
print(MaxDecimalNumber(8))
```

```
print(MaxDecimalNumber(9))
print(MaxDecimalNumber(10))
print(MaxDecimalNumber(11))
print(20*'-')
#=============================================
#====== 2-2, #10 ================================
print('Problem 2-2, #10')
def generatePrintBinary(m,n):
    from queue import Queue
    q = Queue()
    q.put(str(m))#q.put("1")
    while(n > 0):
        n -= 1
        s1 = q.get()
        print(s1)
        s2 = s1
        q.put(s1+"0")
        q.put(s2+"1")

m = input('start of the sequence:')
n = int(input('end of sequence:'))
print('=>')
generatePrintBinary(m,n)
print(20*'-')
#=============================================
#====== 2-3, #13 ================================
print('Problem 2-3, #13')
def RepeatDivision(n):
    ans = ''
    while n != 0:
        r = n % 2
        n = n //2
        ans = str(r) + ans
    print(ans)
RepeatDivision(13)
RepeatDivision(17)
```

```
RepeatDivision(23)
RepeatDivision(30)
RepeatDivision(35)
RepeatDivision(40)
RepeatDivision(49)
RepeatDivision(60)
print(20*'-')
#=============================================
#====== 2-4, #15 ================================
print('Problem 2-4, #15')
def AddBinary(m,n):
    return ((int(m,2)) + (int(n,2)))

if __name__ == '__main__':
    print(AddBinary('10','10'))
    print(AddBinary('10','11'))
    print(AddBinary('100','11'))
    print(AddBinary('111','101'))
    print(AddBinary('1111','111'))
    print(AddBinary('1111','1111'))
print(20*'-')
#=============================================
#====== 2-5, #19 ================================
print('Problem 2-5, #19')
print('2 ways of representing ones complement is by 00000000 and
11111111. ')
print(20*'-')
#=============================================

#=============================================
2-6, #28 ========================================

a) 10011001
2^0 + 2^3 + 2^4 = 1 + 8 + 16 = 25 => -25 because there's a 1 on the
leftmost.
```

b) 01110100
2^2 + 2^4 + 2^5 + 2^6 = 4 + 16 + 32 + 64 = 116

c) 10111111
2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 = 1 + 2 + 4 + 8 + 16 + 32 = 63 ⇒ -63

#==========
#====== 2-6, #29 ==========

a) 0111110000101011
   => 0.111110000101010 * 2^15
   => 15 => Binary Conversion by code above => 00001111
   => 0 is MSB
   => SPF: 000001111111110000101011
b) 100110000011000
   => 1.00110000011000 * 2^14
   => 14 => Binary Conversion by code above => 00001110
   => 1 is MSB
   => SPF: 10000111000110000011000

Below are my results from the code until problems 2-6, #28 and #29, which I worked out by hand and typed.

```
In [2]: runcell(0, 'C:/Users/cedc3/.spyder-py3/temp.py')
Problem 2-2, #6
33
39
42
57
96
253
242
255
--------------------
Problem 2-2, #8
3
7
15
31
63
127
255
511
1023
2047
--------------------
Problem 2-2, #10

start of the sequence:0

end of sequence:7
=>
0
00
01
000
001
010
011
--------------------
Problem 2-3, #13
1101
10001
10111
11110
100011
101000
110001
111100
--------------------
```

```
--------------------
Problem 2-4, #15
4
5
7
12
22
30
--------------------
Problem 2-5, #19
2 ways of representing ones complement is by 00000000 and 11111111.
--------------------
```