Lucia Piner

Cosc 330 HW1

#from file hw1_function.py:

```
#cosc 330 hw 1 functions
def dec_to_bin(xd):
    result = []
    while (xd > 0):
        result.append(xd%2)
        xd = xd//2
    result.reverse()
    return result

def bin_to_dec(xb):
    result = 0
    nums = [int(bit) for bit in str(xb)]
    nums.reverse()
    for i in range(0,len(nums)):
        result = result + nums[i]*2**i
    return result

def max_num(n):
    return 2**n - 1

def sequence(m,n):
    for i in range(m,n+1):
        if i == 0:
            print('[0]\n')
        else:
            print(dec_to_bin(i),"\n")
    return 0

def add(a,b):
    return bin_to_dec(a)+bin_to_dec(b)

def get_value_2_comp(xb):
    result = 0
    nums = [int(bit) for bit in str(xb)]
    nums.reverse()
    for i in range(0,len(nums)-1):
        result = result + nums[i]*2**i
    #sign bit
    result = result - 2**(len(nums)-1)*nums[len(nums)-1]
    return result
    return 0
```

```
if __name__ == "__main__":
    print(dec_to_bin(8))
    print(bin_to_dec(1010))
    print(max_num(3))
    sequence(0,7)
    print(add(101,11))
    print(get_value_2_comp(10011001))


#=====================================================
```

#from file hw1.py:

```
from hw1_functions import *
print('---6---')
print('a: 1100 =',bin_to_dec(1100))
print('b: 1010 =',bin_to_dec(1010))
print('c: 11100 =',bin_to_dec(11100))
print('d: 10000 =',bin_to_dec(10000))
print('e: 10101 =',bin_to_dec(10101))
print('f: 11101 =',bin_to_dec(11101))
print('g: 10111 =',bin_to_dec(10111))
print('h: 11111 =',bin_to_dec(11111))
print()

print('---8---')
n = 2
while (n<=11):
    print('n =',n,":",max_num(n))
    n+=1
print()

print('---10---')
print('0-7:')
sequence(0,7)
print('8-15:')
sequence(8,15)
print('16-31:')
sequence(16,31)
print('32-63:')
sequence(32,63)
print('64-75:')
sequence(64,75)
print()
```

```
print('---13---')
print('a: 15 =',dec_to_bin(15))
print('b: 21 =',dec_to_bin(21))
print('c: 28 =',dec_to_bin(28))
print('d: 34 =',dec_to_bin(34))
print('e: 40 =',dec_to_bin(40))
print('f: 59 =',dec_to_bin(59))
print('g: 65 =',dec_to_bin(65))
print('h: 73 =',dec_to_bin(73))
print()

print('---15---')
print('a: 11+01 =',add(11,1))
print('b: 10+10 =',add(10,10))
print('c: 101+11 =',add(101,11))
print('d: 111+110 =',add(111,110))
print('e: 1001+101 =',add(1001,101))
print('f: 1101+1011 =',add(1101,1011))
print()

print('---19---')
print("either all 0's or all 1's:")
print("0000 = 0 -> 0+0+0+0 = 0")
print("1111 = 0 -> -8+4+2+1 = -1 (+1) = 0")
print()

print('---28---')
#string bc leading 0 is not allowed for decimal numbers in
#python
print('10011001 =',get_value_2_comp('10011001'))
print('01110100 =',get_value_2_comp('01110100'))
print('10111111 =',get_value_2_comp('10111111'))
print()

print('---29---')
print('see photo')
```

29 Each # is in sign-magnitude form
express as single precision floating point #s:

a 0111 1100 0010 1011

↓ $1.(11\ 1100\ 0010\ 1011) \times 2^{14}$

S = 0 → positive

$14 + 127 = 141 \rightarrow 128 + 8 + 4 + 1$

E = 1000 1101

0 1000 1101, 1111 0000 1010 1100 0000 000,

S E M

b 1 0011 0000 0110 00

↓ $-1.(10000\ 01100\ 0) \times 2^{11}$

S = 1 → negative

M

$11 + 127 = 138 \rightarrow 128 + 8 + 2$

E = 1000 1010

1 1000 1010, 1000 0011 0000 0000 0000 000

S E M