Homework Number: 07
Name: Yi Qiao
ECN Login: qiao22
Due Date: 03/07/2019

Code

```
#! /usr/bin/env python3

from BitVector import *
import sys

K = [0x428a2f98d728ae22, 0x7137449123ef65cd, 0xb5c0fbcfec4d3b2f,
 ↪  0xe9b5dba58189dbbc, 0x3956c25bf348b538,
      0x59f111f1b605d019, 0x923f82a4af194f9b, 0xab1c5ed5da6d8118,
       ↪  0xd807aa98a3030242, 0x12835b0145706fbe,
      0x243185be4ee4b28c, 0x550c7dc3d5ffb4e2, 0x72be5d74f27b896f,
       ↪  0x80deb1fe3b1696b1, 0x9bdc06a725c71235,
      0xc19bf174cf692694, 0xe49b69c19ef14ad2, 0xefbe4786384f25e3,
       ↪  0x0fc19dc68b8cd5b5, 0x240ca1cc77ac9c65,
      0x2de92c6f592b0275, 0x4a7484aa6ea6e483, 0x5cb0a9dcbd41fbd4,
       ↪  0x76f988da831153b5, 0x983e5152ee66dfab,
      0xa831c66d2db43210, 0xb00327c898fb213f, 0xbf597fc7beef0ee4,
       ↪  0xc6e00bf33da88fc2, 0xd5a79147930aa725,
      0x06ca6351e003826f, 0x142929670a0e6e70, 0x27b70a8546d22ffc,
       ↪  0x2e1b21385c26c926, 0x4d2c6dfc5ac42aed,
      0x53380d139d95b3df, 0x650a73548baf63de, 0x766a0abb3c77b2a8,
       ↪  0x81c2c92e47edaee6, 0x92722c851482353b,
      0xa2bfe8a14cf10364, 0xa81a664bbc423001, 0xc24b8b70d0f89791,
       ↪  0xc76c51a30654be30, 0xd192e819d6ef5218,
      0xd69906245565a910, 0xf40e35855771202a, 0x106aa07032bbd1b8,
       ↪  0x19a4c116b8d2d0c8, 0x1e376c085141ab53,
      0x2748774cdf8eeb99, 0x34b0bcb5e19b48a8, 0x391c0cb3c5c95a63,
       ↪  0x4ed8aa4ae3418acb, 0x5b9cca4f7763e373,
      0x682e6ff3d6b2b8a3, 0x748f82ee5defb2fc, 0x78a5636f43172f60,
       ↪  0x84c87814a1f0ab72, 0x8cc702081a6439ec,
      0x90befffa23631e28, 0xa4506cebde82bde9, 0xbef9a3f7b2c67915,
       ↪  0xc67178f2e372532b, 0xca273eceea26619c,
      0xd186b8c721c0c207, 0xeada7dd6cde0eb1e, 0xf57d4f7fee6ed178,
       ↪  0x06f067aa72176fba, 0x0a637dc5a2c898a6,
      0x113f9804bef90dae, 0x1b710b35131c471b, 0x28db77f523047d84,
       ↪  0x32caab7b40c72493, 0x3c9ebe0a15c9bebc,
      0x431d67c49c100d4c, 0x4cc5d4becb3e42b6, 0x597f299cfc657e2a,
       ↪  0x5fcb6fab3ad6faec, 0x6c44198c4a475817]

K_bv = [BitVector(intVal = k) for k in K]
```

```
def sha512(input_file, output_file):

    h0 = BitVector(hexstring = '6a09e667f3bcc908')
    h1 = BitVector(hexstring = 'bb67ae8584caa73b')
    h2 = BitVector(hexstring = '3c6ef372fe94f82b')
    h3 = BitVector(hexstring = 'a54ff53a5f1d36f1')
    h4 = BitVector(hexstring = '510e527fade682d1')
    h5 = BitVector(hexstring = '9b05688c2b3e6c1f')
    h6 = BitVector(hexstring = '1f83d9abfb41bd6b')
    h7 = BitVector(hexstring = '5be0cd19137e2179')

    message = None
    with open(input_file, "r") as fp:
        message = fp.read()

    words = [None] * 80
    # padding
    bv = BitVector(textstring=message) + BitVector(bitstring="1")
    msg_len = bv.length() - 1
    bv.pad_from_right((896 - bv.length()) % 1024)
    bv += BitVector(intVal = msg_len, size=128)


    # message scheduling for 1024-bit input block
    for n in range(0, bv.length(), 1024):
        block = bv[n: n+1024]
        words[0:16] = [block[i:i+64] for i in range(0, 1024, 64)]

        for i in range(16,80):

            sigma_0 = (words[i-15].deep_copy() >> 1) ^ (words[i-15].deep_copy()
            ↪  >> 8) ^ (words[i-15].deep_copy().shift_right(7))
            sigma_1 = (words[i-2].deep_copy() >> 19) ^ (words[i-2].deep_copy() >>
            ↪  61) ^ (words[i-2].deep_copy().shift_right(6))
            words[i] = BitVector(intVal=(int(words[i-16]) + int(sigma_0) +
            ↪  int(words[i-7]) + int(sigma_1)) & 0xFFFFFFFFFFFFFFFF, size = 64)

        a,b,c,d,e,f,g,h = h0,h1,h2,h3,h4,h5,h6,h7

        # process 80 rounds
        for i in range(80):
            ch = (e & f) ^ ((~e) & g)
            maj = (a & b) ^ (a & c) ^ (b & c)
            sum_a = ((a.deep_copy()) >> 28) ^ ((a.deep_copy()) >> 34) ^
            ↪  ((a.deep_copy()) >> 39)
            sum_e = ((e.deep_copy()) >> 14) ^ ((e.deep_copy()) >> 18) ^
            ↪  ((e.deep_copy()) >> 41)
            t1 = BitVector(intVal=(int(h) + int(ch) + int(sum_e) + int(words[i])
             ↪  +int(K_bv[i])) & 0xFFFFFFFFFFFFFFFF, size=64)
            t2 = BitVector(intVal=(int(sum_a) + int(maj)) & 0xFFFFFFFFFFFFFFFF,
             ↪  size=64)
            h = g
            g = f
            f = e
            e = BitVector(intVal=(int(d) + int(t1)) & 0xFFFFFFFFFFFFFFFF,
             ↪  size=64)
            d = c
            c = b
            b = a
            a = BitVector(intVal=(int(t1) + int(t2)) & 0xFFFFFFFFFFFFFFFF,
             ↪  size=64)


        # mix back with the contents of the hash buffer
        h0 = BitVector( intVal = (int(h0) + int(a)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h1 = BitVector( intVal = (int(h1) + int(b)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h2 = BitVector( intVal = (int(h2) + int(c)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h3 = BitVector( intVal = (int(h3) + int(d)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h4 = BitVector( intVal = (int(h4) + int(e)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h5 = BitVector( intVal = (int(h5) + int(f)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h6 = BitVector( intVal = (int(h6) + int(g)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )
        h7 = BitVector( intVal = (int(h7) + int(h)) & 0xFFFFFFFFFFFFFFFF, size=64
         ↪  )

    message_hash = h0 + h1 + h2 + h3 + h4 + h5 + h6 + h7

    with open(output_file, "w") as fp:
        fp.write(message_hash.get_bitvector_in_hex())


if __name__ == "__main__":

    if len(sys.argv) != 3:
        print("Usage: ./sha512.py <input file> <output file>")
        sys.exit(1)

    sha512(sys.argv[1], sys.argv[2])
```