3. Exercises - Q1

Exercises from the book using checksum.py

Recommended change so -D option can use other bases:

```
$ diff checksum.py.ORIG checksum.py
48c48
<         values.append(int(t))
---
>         values.append(int(t,0)) # automatic base detection, 0b1010..., 0xabc, etc.
$
```

Q1:

```
$ python ./checksum.py

Decimal:          216        194        107         66
Hex:             0xd8       0xc2       0x6b       0x42
Bin:       0b11011000 0b11000010 0b01101011 0b01000010

Add:      ?
Xor:      ?
Fletcher: ?

--> check by hand

$ python ./checksum.py -c

Decimal:          216        194        107         66
Hex:             0xd8       0xc2       0x6b       0x42
Bin:       0b11011000 0b11000010 0b01101011 0b01000010

Add:              71       (0b01000111)
Xor:              51       (0b00110011)
Fletcher(a,b):    73,196   (0b01001001,0b11000100)
```

```
$ python3
>>> d = [216, 194, 107, 66]
>>> X = d[0]^d[1]^d[2]^d[3]
>>> X
51
>>> bin(X)
'0b110011'
>>> A = d[0]+d[1]+d[2]+d[3]
>>> A
583
>>> A % 256
71
>>> A & 0xff
71
>>> A % 255
73
>>> A0 = d[0]; A1 = A0+d[1]; A2 = A1+d[2]; A3 = A2+d[3]
>>> (A0+A1+A2+A3) % 255
196
>>>
```

```
  Xor            Add


  11011000       11011000
  11000010       11000010
  01101011       01101011
  01000010       01000010
  --------       --------
```