Exercise 2.11

(a)

```
>>> 1+2+3+4+5+6+7
28
```

(b)

```
>>> (65+57+45)/3
55.666666666666664
```

(c)

```
>>> 2**20
1048576
```

(d)

```
>>> 4356//61
71
```

(e)

```
>>> 4356%61
25
```

Exercise 2.12

```
>>> s1 = '-'
>>> s2 = '+'
```

(a)

```
>>> s1 + s2
'-+'
```

(b)

```
>>> s1 + s2 + s1
'-+-'
```

(c)

```
>>> s2 + 2 * s1
'+--'
```

(d)

```
>>> 2 * (s2 + 2 * s1)
```

'+--+--'

(e)

>>> 10 * (s2 + 2 * s1) + s2

'+--+--+--+--+--+--+--+--+--+--+'

(f)

>>> 5 * (s2 + s1 + 3 * s2 + 2 * s1)

'+-+++--+-+++--+-+++--+-+++--+-+++--'

Exercise 2.13

>>> s = 'abcdefghijklmnopqrstuvwxyz'

>>> s[0]

'a'

>>> s[2]

'c'

>>> s[-1]

'z'

>>> s[-2]

'y'

>>> s[-10]

'q'

Exercise 2.14

>>> s = 'goodbye'

(a)

>>> s[0] == 'g'

True

(b)

>>> s[6] == 'g'

False

(c)

```
>>> s[0] == 'g' and s[1] == 'a'
False
(d)
>>> s[-2] == 'x'
False
(e)
>>> s[3] == 'd'
True
(f)
>>> s[0] == s[-1]
False
(g)
>>> s[-1] == 'n' and s[-2] == 'o' and s[-3] == 'i' and s[-4] == 't'
False


# Exercise 2.15
(a)
>>> len('anachronistically') == len('counterintuitive') + 1
True
(b)
>>> 'misinterpretation' < 'misrepresentation'
True
(c)
>>> 'e' not in 'floccinaucinihilipilification'
True
(d)
>>> len('counterrevolution') == len('counter') + len('resolution')
True


# Exercise 2.16
```

(a)

```
>>> a = 6
>>> b = 7
```

(b)

```
>>> c = (a + b)/2
```

(c)

```
>>> inventory = ['paper', 'staples', 'pencils']
```

(d)

```
>>> first = 'John'
>>> middle = 'Fitzgerald'
>>> last = 'Kennedy'
```

(e)

```
>>> fullname = first + ' ' + middle + ' ' + last
```

Exercise 2.17

(a)

```
>>> 17 + (-9) < 10
True
```

(b)

```
>>> len(inventory) > 5 * len(fullname)
False
```

(c)

```
>>> c <= 24
True
```

(d)

```
>>> a < 6.75 < b
True
```

(e)

```
>>> len(first) < len(middle) < len(last)
False
```

(f)

```
>>> len(inventory) == 0 or len(inventory) > 10
False
```

Exercise 2.18

(a)

```
>>> flowers = ['rose', 'bougainvillea', 'yucca', 'marigold', 'daylilly', 'lilly of the valley']
```

(b)

```
>>> 'potato' in flowers
False
```

(c)

```
>>> thorny = [flowers[0], flowers[1], flowers[2]]
```

(d)

```
>>> poisonous = [flowers[-1]]
```

(e)

```
>>> dangerous = thorny + poisonous
```

Exercise 2.19

```
>>> answers = ['Y', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'N', 'N']
```

(a)

```
>>> numYes = answers.count('Y')
>>> numYes
5
```

(b)

```
>>> numNo = answers.count('N')
>>> numNo
6
```

(c)

```
>>> percentYes = numYes*100/len(answers)
>>> percentYes
```

45.45454545454545

(d)

```
>>> answers.sort()
>>> answers
['N', 'N', 'N', 'N', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y']
```

(e)

```
>>> f = answers.index('Y')
>>> f
6
```

```
# Exercise 2.20
>>> s = 'top'
>>> s[-1] + s[1] + s[0]
'pot'
```

```
# Exercise 2.21
>>> s = 'Perkovic'
>>> t = 'Ljubomir'
>>> t[0] + s[0]
'LP'
```

```
# Exercise 2.22
>>> lst = [3, 7, -2, 12]
>>> max(lst) - min(lst)
14
```

```
# Exercise 2.23
>>> monthsL = ['Jan', 'Feb', 'Mar', 'May']
```

```
>>> monthsT = ('Jan', 'Feb', 'Mar', 'May')
(a)
>>> monthsL.insert(3, 'Apr')
>>> monthsL
['Jan', 'Feb', 'Mar', 'Apr', 'May']
>>> monthsT.insert(3, 'Apr')
Traceback (most recent call last):
  File "<pyshell#24>", line 1, in <module>
    monthsT.insert(3, 'Apr')
AttributeError: 'tuple' object has no attribute 'insert'
(b)
>>> monthsL.append('Jun')
>>> monthsL
['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun']
>>> monthsT.append('Jun')
Traceback (most recent call last):
  File "<pyshell#27>", line 1, in <module>
    monthsT.append('Jun')
AttributeError: 'tuple' object has no attribute 'append'
(c)
>>> monthsL.pop()
'Jun'
>>> monthsL
['Jan', 'Feb', 'Mar', 'Apr', 'May']
>>> monthsT.pop()
Traceback (most recent call last):
  File "<pyshell#30>", line 1, in <module>
    monthsT.pop()
AttributeError: 'tuple' object has no attribute 'pop'
(d)
>>> monthsL.pop(1)
'Feb'
```

```
>>> monthsL
['Jan', 'Mar', 'Apr', 'May']
>>> monthsT.pop(1)
Traceback (most recent call last):
  File "<pyshell#34>", line 1, in <module>
    monthsT.pop(1)
AttributeError: 'tuple' object has no attribute 'pop'
(e)
>>> monthsL.reverse()
>>> monthsL
['May', 'Apr', 'Mar', 'Jan']
>>> monthsT.reverse()
Traceback (most recent call last):
  File "<pyshell#37>", line 1, in <module>
    monthsT.reverse()
AttributeError: 'tuple' object has no attribute 'reverse'
(f)
>>> monthsL.sort()
>>> monthsL
['Apr', 'Jan', 'Mar', 'May']
>>> monthsT.sort()
Traceback (most recent call last):
  File "<pyshell#40>", line 1, in <module>
    monthsT.sort()
AttributeError: 'tuple' object has no attribute 'sort'



# Exercise 2.24
>>> grades = ['B','B','F','C','B','A','A','D','C','D','A','A','B']
>>> count = []
>>> count.append(grades.count('A'))
>>> count.append(grades.count('B'))
```

```
>>> count.append(grades.count('C'))
>>> count.append(grades.count('D'))
>>> count.append(grades.count('F'))
>>> count
[4, 4, 2, 2, 1]


# Exercise 2.25
>>> grades = ('B','B','F','C','B','A','A','D','C','D','A','A','B')
>>> count = []
>>> count.append(grades.count('A'))
>>> count.append(grades.count('B'))
>>> count.append(grades.count('C'))
>>> count.append(grades.count('D'))
>>> count.append(grades.count('F'))
>>> count
[4, 4, 2, 2, 1]


# Exercise 2.26
(a)
>>> x = 0
>>> y = 0
>>> x**2 + y**2 <= 10**2
True
(b)
>>> x = 10
>>> y = 10
>>> x**2 + y**2 <= 10**2
False
(c)
>>> x = 6
```

```
>>> y = -6
>>> x**2 + y**2 <= 10**2
True
(d)
>>> x = -7
>>> y = 8
>>> x**2 + y**2 <= 10**2
False


# Exercise 2.27
>>> s[-1] + s[1] + s[0]
(a)
>>> length = 16
>>> angle = 75
>>> import math
>>> height = length * math.sin(math.pi * angle / 180)
>>> height
15.454813220625093
(b)
>>> length = 20
>>> angle = 0
>>> height = length * math.sin(math.pi * angle / 180)
>>> height
0.0
(c)
>>> length = 24
>>> angle = 45
>>> height = length * math.sin(math.pi * angle / 180)
>>> height
16.97056274847714
(d)
```

```
>>> length = 24
>>> angle = 80
>>> height = length * math.sin(math.pi * angle / 180)
>>> height
23.63538607229299
```

Exercise 2.28

```
>>> lst = ['bat', 'dog', 'cat', 'ant']
```

(a)

```
>>> len(lst)//2
2
```

(b)

```
>>> lst[len(lst)//2]
'cat'
```

(c)

```
>>> lst.sort()
>>> lst
['ant', 'bat', 'cat', 'dog']
```

(d)

```
>>> lst.append(lst.pop(0))
>>> lst
['bat', 'cat', 'dog', 'ant']
```

Exercise 2.29

(a)

```
>>> 0 == (1 == 2)
True
```

(b)

```
>>> 2 + (3 == 4) + 5 == 7
True
```

(c)

```
>>> (1 < -1) == (3 > 4)
True
>>>
```

Exercise 2.30

```
>>> list('computer')
['c', 'o', 'm', 'p', 'u', 't', 'e', 'r']
```

When given a string input, the list constructor returns a list whose items are the characters, in order from left to right, of the string.

Exercise 2.31

List method extend() takes as input a list and appends its items to list lst. Method copy() returns a copy of list lst. Method clear() empties list lst.