1 Exercises

Exercise 1. Consider the following code fragment:

```
a = [0]
for i in range(1, 6):
    a += [a[i - 1] + i]
```

a. What is the value of `a[5]`?

b. What is the value of `sum(a)`?

Exercise 2. What does the following code fragment write?

```
a = ["it", "was", "the", "best", "of", "times", "it", "was", "the", "worst", "of", "times"]
x = 0
y = 0
for v in a:
    x += 1
    y += len(v)
stdio.writeln(str(x) + " " + str(y))
```

Exercise 3. What does the following code fragment write?

```
a = [1, 2, 3, 4, 5]
b = a
b[2] = 0
stdio.writeln(sum(a))
```

Exercise 4. Suppose `a = ["mercury", "venus", "earth", "mars", "jupiter", "saturn", "uranus", "neptune"]`. What are the values of the following expressions?

a. `len(a)`

b. `a[2]`

c. `a[3:]`

d. `a[:3]`

e. `a[-2]`

f. `a[-2:]`

g. `a[:-2]`

h. `a[:]`

Exercise 5. What does the following code fragment write?

```
a = [[1, 2, 3], [2, 3, 4], [3, 4, 5]]
x = 0
for i in range(len(a)):
    for j in range(len(a[0])):
        x += a[i][j]
stdio.writeln(x)
```

Exercise 6. What does the following code fragment write?

```
a = stdarray.create1D(4, None)
for i in range(len(a)):
    a[i] = stdarray.create1D(i + 1, 2)
stdio.writeln(sum(a[3]))
```

Exercise 7. Consider the following program `mystery.py`:

mystery.py

```
import stdarray
import stdio
import sys

n = int(sys.argv[1])
a = stdarray.create2D(n, n, "-")
for i in range(n):
    for j in range(n):
        if i == j or i + j == n - 1:
            a[i][j] = "*"
for i in range(n):
    for j in range(n):
        if j == n - 1:
            stdio.writeln(a[i][j])
        else:
            stdio.write(str(a[i][j]) + " ")
```

a. What does the program write in general?

b. What does the program write when run with the command-line argument $n = 5$?

Exercise 8. Write a program called `die_rolls.py` that accepts n (int) and *trials* (int) as command-line arguments, rolls a fair n-sided die *trials* times, and reports the number of times each of the n values was rolled. For example

~/workspace/ipp/programs

```
$ python3 die_rolls.py 6 100
1 -> 19 times
2 -> 16 times
3 -> 12 times
4 -> 19 times
5 -> 15 times
6 -> 19 times
```

Exercise 9. What do the following code fragments write?

a.

```
x = (["a", "b", "c"], [1, 2, 3, 4, 5])
stdio.writeln(len(x) + len(x[0]) + len(x[1]))
```

b.

```
x = set("panama")
y = set("canal")
stdio.writeln(x | y)
stdio.writeln(x & y)
stdio.writeln(x - y)
stdio.writeln(y - x)
stdio.writeln(x ^ y)
```

c.

```
x = {"a": 1, "b": 2, "c": 3}
y = "a" * x["a"] + "b" * x["b"] + "c" * x["c"]
stdio.writeln(y)
```

Exercise 10. What do the following code fragments write?

a.

```
for x, y in enumerate(range(1, 10, 2)):
    stdio.writeln(str(x) + ":" + str(y * y))
```

b.

```
w = 0
for x, y, z in zip([1, 2, 3], [4, 5, 6], [7, 8, 9]):
    w += x * y * z
stdio.writeln(w)
```

c.

```
x = ["it", "was", "the", "best", "of", "times", "it", "was", "the", "worst", "of", "times"]
for v in reversed(sorted(x)):
    stdio.writeln(v)
```

2 Solutions

Solution 1.

a. `15`

b. `35`

Solution 2.

```
12 39
```

Solution 3.

```
12
```

Solution 4.

a. `8`

b. `"earth"`

c. `["mars", "jupiter", "saturn", "uranus", "neptune"]`

d. `["mercury", "venus", "earth"]`

e. `"uranus"`

f. `["uranus", "neptune"]`

g. `["mercury", "venus", "earth", "mars", "jupiter", "saturn"]`

h. `["mercury", "venus", "earth", "mars", "jupiter", "saturn", "uranus", "neptune"]`

Solution 5.

```
27
```

Solution 6.

```
8
```

Solution 7.

a. The program writes an $n \times n$ matrix in which the diagonal elements are stars and the off-diagonal elements are dashes.

b.

```
* - - - *
- * - * -
- - * - -
- * - * -
* - - - *
```

Solution 8.

die_rolls.py

```
import stdarray
import stdio
import stdrandom
import sys

n = int(sys.argv[1])
trials = int(sys.argv[2])
rolls = stdarray.create1D(n + 1, 0)
for i in range(trials):
    v = stdrandom.uniformInt(1, n + 1)
    rolls[v] += 1
for i in range(1, n + 1):
    stdio.writeln(str(i) + " -> " + str(rolls[i]) + " times")
```

Solution 9.

a.

```
10
```

b.

```
abbccc
```

c.

```
{"a", "p", "m", "c", "l", "n"}
{"a", "n"}
{"p", "m"}
{"c", "l"}
{"p", "m", "c", "l"}
```

Solution 10.

a.

```
0:1
1:9
2:25
3:49
4:81
```

b.

```
270
```

c.

```
worst
was
was
times
times
the
the
of
of
it
it
best
```