### Contents

1. Building a Computer 2
2. Your First Program 3
3. Built-in Types of Data 4
4. Control Flow 5
5. Collections 7
6. Input and Output 11
7. Case Study: PageRank Algorithm 12
8. Defining Functions 13
9. Modules and Clients 15
10. Recursion 16
11. Case Study: Percolation Problem 17
12. Using Data Types 17
13. Creating Data Types 19
14. Designing Data Types 20
15. Case Study: N-body Problem 23
16. Searching and Sorting 24
17. Stacks and Queues 25
18. Case Study: Small-world Problem 26
19. Limits of Computation 27
1 Building a Computer

Problems

Problem 1. What is the range of non-negative integers that can be represented using 16 bits.

Problem 2. Consider the decimal number 245.
   a. Express it as an 8-digit binary number.
   b. Express it as a 3-digit octal (base 8) number.

Problem 3. Consider the decimal numbers 77 and 84.
   a. Express the numbers as 8-digit binary numbers.
   b. Add the two numbers in binary.
   c. Express the negative of the result as an 8-digit binary number.

Problem 4. Suppose we are using 8 decimal digits to represent our numbers, the first six of which are for the fractional part and last two for the exponent. What real number does 27182801 represent?

Problem 5. Consider a 1-bit full adder (FA) with inputs \(x\), \(y\), and \(c_{in}\) (carry-in), and outputs \(z\) and \(c_{out}\) (carry-out).
   a. Give the truth table for FA.
   b. Use the minterm expansion principle to derive Boolean functions \(z = f(x, y, c_{in})\) and \(c_{out} = g(x, y, c_{in})\) that respectively express the output \(z\) and \(c_{out}\) in terms of the inputs \(x\), \(y\), and \(c_{in}\).

Problem 6. Consider an 8-bit computer with four instructions: add (0), sub (1), mul (2), and div (3). Assume the computer has four registers, numbered 0 through 3.
   a. What is the 8-bit binary code for the instruction \texttt{mul 3 1 2} (multiply the values in registers 1 and 2, and store the result in register 3)?
   b. What is the instruction corresponding to the 8-bit binary code 11011011?

Solutions

Solution 1. \(0, 1, 2, 3, \ldots, 2^{16} - 1 = 65535\)

Solution 2.
   a. \(245_{10} = 11110101_2\)
   b. \(245_{10} = 365_8\)

Solution 3.
   a. \(77_{10} = 01001111_2\) and \(84_{10} = 01010100_2\)

   \[
   \begin{array}{cccc}
   1 & 1 & 1 & 1 \\
   0 & 1 & 0 & 0 \\
   \hline
   1 & 0 & 1 & 0
   \end{array}
   \]

   b. \(+\) \[\begin{array}{cccc}
   0 & 1 & 0 & 1 \\
   0 & 1 & 0 & 1 \\
   \hline
   1 & 0 & 1 & 0
   \end{array}\]

   c. \(-161_{10} = 01011111_2\)

Solution 4. 2.71828 (Euler’s constant up to 5 decimal places)

Solution 5.
<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$c_{in}$</th>
<th>$z$</th>
<th>$c_{out}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>a.</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

b. $z = f(x, y, c_{in}) = \overline{x}y\overline{c}_{in} + xy\overline{c}_{in} + xyc_{in}$; $c_{out} = g(x, y, c_{in}) = \overline{x}y\overline{c}_{in} + xy\overline{c}_{in} + xyc_{in}$

Solution 6.

a. mul 3 1 2 $\Rightarrow$ 10110110

b. 11011011 $\Rightarrow$ div 1 2 3 (divide the value in register 2 by the value in register 3 and store the result in register 1)

2 Your First Program

Problems

Problem 1. Consider the following program mystery.py:

```python
# mystery.py
import stdio
import sys
a = sys.argv
```

What does the program write when it is run as follows?

$ python mystery.py Alice Bob Carol Dan Eve

Problem 2. a. Write a program date.py that takes three command-line arguments $d$, $m$, and $y$ representing day, month, and year, and writes the date using the “$y/m/d$” format.

b. What is the command for running date.py on the inputs $d = 14$, $m = 3$, and $y = 1879$?

Solutions

Solution 1. Hi Dan, Bob, and Alice!

Solution 2.

a. # date.py

```python
import stdio
import sys
```

b. python date.py 14 3 1879
3 Built-in Types of Data

**PROBLEMS**

**Problem 1.** What is the value and type of each of the following expressions?

a. '1' + '-' + '1'

b. 'This parrot would not voom if you put ' + str(4) + ' million volts through it!'

c. '42' * 3

d. int('42') * 3

e. 1 - 1 - 1 - 1

f. 3 / 2.0 + 2 * 5

g. 3 // 2 + 2 * 5

h. 3.14159 + int(math.pi)

i. 8 <= 2 or 2e8 <= 8e2

j. 5 + int(random.random() * 5)

**Problem 2.** Consider the following program mystery.py:

```
import stdio
import sys

a = int(sys.argv[1])
b = int(sys.argv[2])
c = int(sys.argv[3])
stdio.writeln(a ** 2 == b ** 2 + c ** 2 or b ** 2 == a ** 2 + c ** 2 or c ** 2 == a ** 2 + b ** 2)
```

a. What does the program write in general?

b. What does the program write when run with command-line arguments 1, 2, and 3?

c. What does the program write when run with command-line arguments 3, 4, and 5?

**Problem 3.** a. Write a program far2cen.py that takes a float \( F \), representing the temperature in Fahrenheit, as command-line argument and writes its Celsius equivalent \( C \), calculated using the formula \( C = \frac{5}{9}(F - 32) \).

b. How would you run the above program on the terminal to convert \( 42 ^\circ F \) to \( ^\circ C \)?

**Solutions**

**Solution 1.**

a. '1 - 1' (str)

b. 'This parrot would not voom if you put 4 million volts through it!' (str)

c. '424242' (str)

d. 126 (int)

e. -2 (int)

f. 11.5 (float)
g. 11 (int)  

h. 6.14159 (float)  

i. False (bool)  

j. A number from the interval [5, 10) (int)  

**Solution 2.**  
a. Reads three command-line arguments \(a, b,\) and \(c\) as integers and prints \texttt{True}\ if the square of any one of them is the sum of the squares of the other two, and \texttt{False}\ otherwise.  
b. \texttt{False}  
c. \texttt{True}  

**Solution 3.**  
a. \# far2cen.py  
   ```python  
   import stdio  
   import sys  
   F = float(sys.argv[1])  
   C = (F - 32) * 5 / 9  
   stdio.writeln(C)  
   ```  
b. \$ python far2cen.py 42

**4 Control Flow**

**Problems**

**Problem 1.** Consider the following code fragment:  
```python  
if (m >= 1 and m <= 5):  
    stdio.write('Spring ')  
elif (m >= 6 and m <= 8):  
    stdio.write('Summer ')  
else:  
    stdio.write('Fall ')  
stdio.writeln(str(y)[2:])  
```  
What does the program write when \(m\) and \(y\) take on the following values?  

a. \(m = 10\) and \(y = 2016\)  
b. \(m = 5\) and \(y = 2017\)  
c. \(m = 6\) and \(y = 2017\)  

**Problem 2.** What does the following code fragment write?  
```python  
i = 9  
while i >= 0:  
    stdio.writeln(str(i) + ' ' + str(2 ** i))  
i -= 2  
```  
**Problem 3.** What are the arithmetic progressions returned by the following calles to \texttt{range()}?  

a. \texttt{range(-5)}  
b. \texttt{range(5)}  

---

5 of 27
c. range(3, 10)

d. range(3, 10, 2)

e. range(5, -5, -1)

**Problem 4.** What does the following code fragment write?

```python
for i in range(3, 40, 4):
    if i % 5 == 0:
        stdio.writeln(i)
```

**Problem 5.** What does the following code fragment write?

```python
i = 1
for c in 'hello':
    stdio.writeln(c * i)
    i += 1
```

**Problem 6.** What does the following code fragment write?

```python
for i in range(5):
    for j in range(6):
        if j == 5:
            stdio.writeln(i + j)
        else:
            stdio.write(str(i + j) + '-

```

**Problem 7.** Implement a program `sum_of_ints.py` that takes an integer \( n \) as command-line argument and writes the value of the sum \( 1 + 2 + 3 + \cdots + (n-1) + n \).

**Problem 8.** Implement a program `matrix.py` that takes integers \( n \) and \( k \) as command-line arguments and writes an \( n \)-by-\( n \) matrix in which the elements below the main diagonal are all zeros and the rest of the elements have the value \( k \). The elements of the matrix must be separated by a single space and each row must end with a newline character at the end. For example

```bash
$ python matrix.py 5 2
2 2 2 2 2
0 2 2 2 2
0 0 2 2 2
0 0 0 2 2
0 0 0 0 2
```

**Solutions**

**Solution 1.**

a. Fall 16

b. Spring 17

c. Summer 17

**Solution 2.**

```
9 512
7 128
5 32
3 8
1 2
```

**Solution 3.**

a. []

b. [0, 1, 2, 3, 4]
c. [3, 4, 5, 6, 7, 8, 9]
d. [3, 5, 7, 9]
e. [5, 4, 3, 2, 1, 0, -1, -2, -3, -4]

Solution 4.
15
35

Solution 5.

h
ee
lll
llll
ooooo

Solution 6.
0-1-2-3-4-5
1-2-3-4-5-6
2-3-4-5-6-7
3-4-5-6-7-8
4-5-6-7-8-9

Solution 7.

# sum_of_ints.py
import stdio
import sys

n = int(sys.argv[1])
total = 0
for i in range(1, n + 1):
    total += i
stdio.writeln(total)

Solution 8.

# matrix.py
import stdio
import sys

n = int(sys.argv[1])
k = int(sys.argv[2])
for i in range(n):
    for j in range(n):
        e = 0 if i > j else k
        if j == n - 1:
            stdio.writeln(e)
        else:
            stdio.write(str(e) + ', ')

5 Collections

Problems

Problem 1. Consider the following code fragment:

```python
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 \( \text{sum}(a) \)?

**Problem 2.** What does the following code fragment write?

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

**Problem 3.** What does the following code fragment write?

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

**Problem 4.** Suppose \( a = ['mercury', 'venus', 'earth', 'mars', '.jupiter', 'saturn', 'uranus', 'neptune'] \). What are the values of the following expressions?

a. \( \text{len}(a) \)

b. \( a[2] \)

c. \( a[3:] \)

d. \( a[3:] \)

e. \( a[:] \)

f. \( a[-5] \)

g. \( a[:5] \)

h. \( a \) is \( a[:] \)

i. \( a \) is \( a[:] \)

**Problem 5.** What does the following code fragment write?

```python
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)
```

**Problem 6.** What does the following code fragment write?

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

**Problem 7.** What are the values of the following expressions?

a. \( \text{set('panama')} \& \text{set('canal')} \)

b. \( \text{set('panama')} - \text{set('canal')} == \text{set('panama')} - \text{set('canal')} | \text{set('canal')} - \text{set('panama')} \)

c. \( \text{sum([x * x for x in [1, 2, 3, 4, 5]])} \)

d. \( \text{sum([x * x for x in [1, 2, 3, 4, 5] if x % 2 != 0]} \)
Problem 8. Consider the following program `mystery.py`:

```python
# 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`?

Problem 9. Write a program `die_rolls.py` that takes command-line arguments `n` and `T` as integers, rolls a fair `n`-sided die `T` times, and reports the number of times each of the `n` values was rolled. For example

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

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. ['mercury', 'venus', 'earth', 'mars', 'jupiter', 'saturn', 'uranus', 'neptune']

f. 'mars'

g. ['mercury', 'venus', 'earth']

h. True

i. False

Solution 5. 27

Solution 6. 8

Solution 7.

a. 'a', 'n'

b. True

c. 55

d. 35

e. [[[1, 4], [1, 5]], [[2, 4], [2, 5]], [[3, 4], [3, 5]]]

f. {0: 1, 1: 2, 2: 4, 3: 8, 4: 16, 5: 32}

g. [4, 10, 18]

h. {'a': 10, 'b': 8, 'c': 4, 'd': 6, 'r': 9}

i. 55

Solution 8.

a. The program writes an n-by-n matrix where the diagonal elements are starts (*) and the off-diagonal elements are dashes (-).

b. * - - - *
   - * - * -
   - - * - -
   - * - * -
   * - - - *

Solution 9.

```python
# die_rolls.py
import stdarray
import stdio
import stdrandom
import sys

n = int(sys.argv[1])
T = int(sys.argv[2])
rolls = stdarray.create1D(n + 1, 0)
for i in range(T):
    v = stdrandom.uniformInt(1, n + 1)
    rolls[v] += 1
for i in range(1, n + 1):
    stdio.writeln(str(i) + ' -> ' + str(rolls[i]) + ' times')
```
6 Input and Output

Problems

Problem 1. What does the following code fragment write?

```python
r = 5
c = 2 * math.pi * r
a = math.pi * r ** 2
stdio.writeln('radius = %.2f, circumference = %.2f, area = %.2f
', r, c, a)
```

Problem 2. Write a program `randomints.py` that takes command-line arguments `n`, `a`, and `b` as integers and writes `n` random integers (one per line) from the range `[a, b]`. For example

```
$ python randomints.py 5 100 1000
376
103
220
395
524
```

Problem 3. Write a program `stats.py` that reads integers from standard input, and computes and writes their mean, variance, and standard deviation, each up to 3 decimal places. For example

```
$ python stats.py
1
2
3
4
5
<ctrl-d>
mean = 3.000 , var = 2.000 , std = 1.414
```

Problem 4. a. What is the command for generating 10000 random integers from the interval [1, 24] using the `randomints.py` program from Problem 2?

b. What is the command for generating 10000 random integers from the interval [1, 24] and saving the output in a file called `numbers.txt`?

c. What is the command for using the `stats.py` program from Problem 3 to calculate stats for the numbers in `numbers.txt`?

d. What is the command to perform the last two tasks in one shot?

Solutions

Solution 1. radius = 5.00, circumference = 31.42, area = 78.54

Solution 2.

```python
# randomints.py
import stdio
import stdrandom
import sys

n = int(sys.argv[1])
a = int(sys.argv[2])
b = int(sys.argv[3])
for i in range(n):
    r = stdrandom.uniformInt(a, b + 1)
    stdio.writeln(r)
```

Solution 3.
# stats.py

```python
import stdio

a = stdio.readInts()
mean = sum(a) / len(a)
var = 0.0
for v in a:
    var += (v - mean)**2
var /= len(a)
std = var ** 0.5
stdio.writeln(' mean = %.3f, var = %.3f, std = %.3f
', mean, var, std)
```

### Solution 4.

a. `python randomints.py 10000 1 24`

b. `python randomints.py 10000 1 24 > numbers.txt`

c. `python stats.py < numbers.txt`

d. `python randomints.py 10000 1 24 | python stats.py`

### 7 Case Study: PageRank Algorithm

#### Problems

**Problem 1.** Consider a 4-page world-wide web represented by the following file:

```
4
0 1
1 2 1 3
2 0
3 2 3 2
```

a. Assuming the 90-10 rule, what is the transition matrix $P$ for the 4-page web?

b. What are the dimensions of $P$?

c. What is initial vector $v_0$ of probabilities if the random surfer starts on page 3?

d. What are the vectors $v_1$, $v_2$, and $v_3$ of probabilities of the random surfer being on each of the 4 pages after 1, 2, and 3 moves?

e. List the pages in reverse (descending) order of their ranks if after 20 moves the vector $v_{20}$ of probabilities are $v_{20} = [0.28348256, 0.27920182, 0.28673482, 0.15058079]$.

#### Solutions

**Solution 1.**

a. $P = \begin{bmatrix}
0.025 & 0.925 & 0.025 & 0.025 \\
0.025 & 0.025 & 0.475 & 0.475 \\
0.925 & 0.025 & 0.025 & 0.025 \\
0.025 & 0.025 & 0.925 & 0.025
\end{bmatrix}$

b. $4 \times 4$

c. $v_0 = [0\ 0\ 0\ 1]$
d. $v_1 = v_0 P = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0.025 & 0.025 & 0.475 & 0.475 \\ 0.925 & 0.025 & 0.025 & 0.025 \\ 0.025 & 0.025 & 0.925 & 0.025 \end{bmatrix} = \begin{bmatrix} 0.025 & 0.025 & 0.925 & 0.025 \end{bmatrix}$

$v_2 = v_1 P = \begin{bmatrix} 0.025 & 0.025 & 0.925 & 0.025 \\ 0.025 & 0.025 & 0.475 & 0.475 \\ 0.925 & 0.025 & 0.025 & 0.025 \\ 0.025 & 0.025 & 0.925 & 0.025 \end{bmatrix} = \begin{bmatrix} 0.8575 & 0.0475 & 0.05875 & 0.03625 \end{bmatrix}$

$v_3 = v_2 P = \begin{bmatrix} 0.8575 & 0.0475 & 0.05875 & 0.03625 \end{bmatrix} \begin{bmatrix} 0.025 & 0.925 & 0.025 & 0.025 \\ 0.025 & 0.025 & 0.475 & 0.475 \\ 0.925 & 0.025 & 0.025 & 0.025 \\ 0.025 & 0.025 & 0.925 & 0.025 \end{bmatrix} = \begin{bmatrix} 0.077875 & 0.79675 & 0.079 & 0.046375 \end{bmatrix}$

e. 2, 0, 1, 3

8 Defining Functions

Problems

Problem 1. Consider the following functions:

```python
def f(x):
    return x + 1

def g(x):
    return x ** 2

def h(x):
    return x % 5
```

a. What does $f(5)$ return?
b. What does $g(f(5))$ return?
c. What does $h(g(f(5)))$ return?
d. What does $f(g(h(17)))$ return?

Problem 2. Consider the following function:

```python
def f(x, y, n = 1):
    return x ** n + y ** n
```

a. What does $f(2, 3)$ return?
b. What does $f(2, 3, 2)$ return?
c. What does $f(n = 3, y = 2, x = 3)$ return?

d. What does $f(g(h(17)))$ return?

Problem 3. What does the following code fragment write?

```python
def duplicate(s):
    return s + s

s = 'Hello '
s = duplicate(s)
t = 'Bye'
t = duplicate(duplicate(duplicate(t)))
stdio.writeln(s + t)
```
Problem 4. Consider the following code fragment:

```python
a = filter(lambda x: x % 7 == 0, range(1, 28))
```

a. What is the value of `a`?

b. What does `sum(a)` return?

Problem 5. Consider the following code fragment:

```python
a = map(lambda x: x + 2, range(1, 6))
b = reduce(lambda x, y: x + y, a)
```

a. What is the value of `a`?

b. What is the value of `b`?

Problem 6. Consider the following program `mystery.py`:

```python
# mystery.py
import stdio
import sys

def f(a = 1.0, b = 1.0, c = 1.0):
    return lambda x: a * x ** 2 + b * x + c

def main():
    a = float(sys.argv[1])
    b = float(sys.argv[2])
    c = float(sys.argv[3])
    x = float(sys.argv[4])
    stdio.writeln(f(a, b, c)(x))

if __name__ == '__main__':
    main()
```

a. What does the program write in general?

b. What does the program write when run with the command-line arguments `a = 0`, `b = 2`, `c = 5`, and `x = 2`?

c. What is the value of `y` in the following interactive Python session?

```python
>>> import mystery
>>> y = mystery.f()(3)
```

Solutions

Solution 1.

a. 6

b. 36

c. 1

d. 5

Solution 2.

a. 5

b. 13

c. 35

Solution 4.

a. [7, 14, 21]

b. 42

Solution 5.

a. [3, 4, 5, 6, 7]

b. 25

Solution 6.

a. The program takes four floats \(a, b, c\), and \(x\) as command-line arguments and writes the value of the quadratic equation \(ax^2 + bx + c\).

b. 9.0

c. 13.0

9 Modules and Clients

Problems

Problem 1. Implement a module called `logic.py` that supports the following API, along with a suitable test client `_main()` that tests all the functions in the API:

<table>
<thead>
<tr>
<th>function</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>any(a)</code></td>
<td>returns True if any of the entries in the boolean list (a) is True, and False otherwise</td>
</tr>
<tr>
<td><code>all(a)</code></td>
<td>returns True if all of the entries in the boolean list (a) are True, and False otherwise</td>
</tr>
</tbody>
</table>

Solutions

Solution 1.

```python
# logic.py

def any(a):
    for v in a:
        if v:
            return True
    return False

def all(a):
    for v in a:
        if not v:
            return False
    return True

def _main():
    import stdio

    x = [True, True, False, True]
    y = [True, True, True, True]
    stdio.writeln(any(x))
    stdio.writeln(all(x))
    stdio.writeln(any(y))
    stdio.writeln(all(y))

    if __name__ == "__main__":
        _main()
```
10 Recursion

PROBLEMS

Problem 1. Consider the following recursive function:

```python
def mystery(a, b):
    if b == 0:
        return 0
    if a == 0:
        return mystery(b - 1, a)
    return b + mystery(b, a - 1)
```

a. What is the value returned by the call `mystery(10, 0)`?
b. What is the value returned by the call `mystery(0, 10)`?
c. What is the value returned by the call `mystery(3, 7)`?
d. What is the value returned by the call `mystery(10, 3)`?
e. What is the value returned by the call `mystery(200, 300)`?
f. What does the function `mystery()` compute in general about `a` and `b`?

Problem 2. Consider the function $S(n) = 1^2 + 2^2 + 3^2 + \cdots + n^2$, where $n$ is a positive integer.

a. What is the value of $S(5)$?
b. Provide a recursive definition for $S(n)$.
c. Implement a function $S(n)$ using recursion, such that it computes and returns $S(n)$
d. Trace the function call $S(5)$.

SOLUTIONS

Solution 1.

a. 0
b. 0
c. 21
d. 30
e. 60000
f. The product $ab$.

Solution 2.

a. 55
b. $S(n) = \begin{cases} n^2 + S(n - 1) & \text{if } n > 1, \\ 1 & \text{if } n = 1. \end{cases}$
c. ```python
def S(n):
    return 1 if n == 1 else n * n + S(n - 1)
```
d. $S(5)$
   $S(4)$
   $S(3)$
   $S(2)$
   $S(1)$
   return 1
   return 2 * 2 + 1 = 5
   return 3 * 3 + 5 = 14
   return 4 * 4 + 14 = 30
   return 5 * 5 + 30 = 55

11 Case Study: Percolation Problem

PROBLEMS

Problem 1. Consider a percolation system represented by the following Boolean matrix:

\[
\begin{array}{cccccccc}
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
\end{array}
\]

a. What’s the size of the system?
b. What’s the vacancy probability of the system?
c. Does the system percolate vertically?
d. Does the system percolate in general?

SOLUTIONS

Solution 1.

a. 8 × 8
b. 31/64
c. No
d. Yes

12 Using Data Types

PROBLEMS

Problem 1. If s is a string, s.upper() returns a copy of s converted to uppercase.

a. What does the following code fragment write?
   ```python
   s = 'Hello World'
   s.upper()
   stdio.writeln(s[6:11])
   ```

b. What does the following code fragment write?
   ```python
   s = 'Hello World'
   s = s.upper()
   stdio.writeln(s[6:11])
   ```
Problem 2. Suppose we have a user-defined data type called Circle that represents a circle of radius $r$ centered at $(h, k)$ and supports the following API:

<table>
<thead>
<tr>
<th>method</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circle(h, k, r)</td>
<td>constructs a circle $c$ of radius $r$ centered at $(h, k)$; when no arguments are given, $c$ is a unit circle centered at the origin</td>
</tr>
<tr>
<td>c.area()</td>
<td>returns the area of $c$</td>
</tr>
<tr>
<td>c.contains(x, y)</td>
<td>returns True if $c$ contains† $(x, y)$ and False otherwise</td>
</tr>
<tr>
<td>c &lt; d</td>
<td>returns True if $c$ is area-wise smaller than $d$, and False otherwise</td>
</tr>
<tr>
<td>c == d</td>
<td>returns True if $c$ and $d$ represent the same circle, and False otherwise</td>
</tr>
<tr>
<td>str(c)</td>
<td>returns a string representation of $c$, as $(h, k, r)$</td>
</tr>
</tbody>
</table>

† A point $(x, y)$ is contained in a circle of radius $r$ centered at $(h, k)$ if $(x - h)^2 + (y - k)^2 \leq r^2$

a. Is the Circle data type immutable?

b. How do you create a Circle object $c_1$ representing a circle centered at $(1, 1)$ and having radius 2?

c. How do you create a Circle object $c_2$ representing a unit circle centered at the origin?

d. How do you obtain the area of $c_1$?

e. How do you check if the point $(1.2, 2.2)$ is contained in $c_1$?

f. How do you compare the areas of two circles represented by Circle objects $c$ and $d$ without invoking the area() method explicitly? What does the code translate to internally?

g. How do you check if two Circle objects $c$ and $d$ represent the same circle? What does the code translate to internally?

h. How do you obtain the string representation of $c_1$? What does the code translate to internally?

i. Provide a list comprehension that creates a list $a$ of 100 Circle objects, each representing a circle centered at the origin and having a random radius from the interval $[0, 1)$.

j. Provide an expression that uses map and reduce to calculate the sum of the areas of the circles stored in the list $a$ from the previous part.

Problem 3. Write a client program filter.py that accepts three floats $h$, $k$, and $r$ as command-line arguments, creates a Circle object $c$ representing a circle centered at $(h, k)$ and having radius $r$, reads in pairs $(x, y)$ of floats from standard input representing points on a 2D plane, and writes the fraction of points that fall inside the circle $c$. For example

```
$ python filter.py 0 0 3
1 2
3 4
1 5
1 3
<ctrl-d>
0.25
```

Solutions

Solution 1.

a. World

b. WORLD

Solution 2.

a. Yes

b. $c_1 = \text{Circle}(1, 1, 2)$
c. \( c_2 = \text{Circle()} \) or \( c_2 = \text{Circle}(0, 0, 1) \)

d. \( c_1.\text{area()} \)

e. \( c_1.\text{contains}(1.2, 2.2) \)

f. \( c < d \) which translates to \( c.\text{__lt__}(d) \) internally

g. \( c == d \) which translates to \( c.\text{__eq__}(d) \) internally

h. \( \text{str}(c_1) \) which translates to \( c_1.\text{__str__}() \) internally

i. \( \text{circles} = \left[ \text{Circle}(r = \text{random.random}) \right] \text{for} \ i \ \text{in} \ \text{range}(100) \]

j. \( \text{reduce} (\lambda x, y: x + y, \text{map}(\lambda x: x.\text{area()}, \text{circles})) \)

**Solution 3.**

```python
# filter.py
import stdio
import sys
from circle import Circle

def main():
    h = float(sys.argv[1])
    k = float(sys.argv[2])
    r = float(sys.argv[3])
    c = Circle(h, k, r)
    total, inside = 0, 0
    while not stdio.isEmpty():
        x = stdio.readFloat()
        y = stdio.readFloat()
        total += 1
        inside += 1 if c.contains(x, y) else 0
    stdout.writeln(1.0 * inside / total)

if __name__ == '__main__':
    main()
```

### 13 Creating Data Types

**Problems**

**Problem 1.** Implement the `Circle` data type from Problem 2 of Section 3.1, along with a suitable client `_main()` that tests all the methods in the API.

**Solutions**

**Solution 1.**

```python
# circle.py
import math

class Circle:
    def __init__(self, h=0.0, k=0.0, r=1.0):
        self._h = h
        self._k = k
        self._r = r

    def area(self):
        return math.pi * self._r ** 2

    def contains(self, x, y):
```
def _lt_(self, other):
    return self.area() < other.area()

def _eq_(self, other):
    return self._h == other._h and self._k == other._k and
    self._r == other._r

def __str__(self):
    return '(' + str(self._h) + ', ' + str(self._k) + ', ' + str(self._r) + ')

if __name__ == '__main__':
    _main()

14 Designing Data Types

PROBLEMS

Problem 1. Reimplement the Circle data type from Problem 2 of Section 3.1, but this time representing a circle internally using the coordinates \((x, y)\) of the lower left corner of the square that inscribes the circle and has side length \(s\).

Problem 2. Implement, along with a suitable test client, a comparable data type called Color, that represents a color in terms of its red, green, and blue components, and supports the following API:

<table>
<thead>
<tr>
<th>method</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color(r=0, g=0, b=0)</td>
<td>construct a Color object (c) given its red, green, and blue components as integers from the interval ([0, 255])</td>
</tr>
<tr>
<td>c.getRed()</td>
<td>the red component of (c)</td>
</tr>
<tr>
<td>c.getGreen()</td>
<td>the green component of (c)</td>
</tr>
<tr>
<td>c.getBlue()</td>
<td>the blue component of (c)</td>
</tr>
<tr>
<td>c.luminosity()</td>
<td>the luminosity of (c) calculated as (0.299r + 0.587g + 0.114b)</td>
</tr>
<tr>
<td>c + d</td>
<td>a new color whose red, green, and blue components are the average values of the corresponding components of (c) and (d)</td>
</tr>
<tr>
<td>c == d</td>
<td>do (c) and (d) represent the same color?</td>
</tr>
<tr>
<td>cmp(c, d)</td>
<td>-1, 0, or 1 depending on whether (c)’s luminosity is less than, equal to, or greater than (d)’s luminosity</td>
</tr>
<tr>
<td>str(c)</td>
<td>string representation of (c) in ((r, g, b)) format</td>
</tr>
</tbody>
</table>

Problem 3. Implement, along with a suitable test client, an iterable data type called RandomColors, that can be used to build and iterate over a collection of random Color objects. The data type must support the following API:

<table>
<thead>
<tr>
<th>method</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RandomColors(n)</td>
<td>an iterable object (r) for iterating over (n) random Color objects</td>
</tr>
<tr>
<td>iter(r)</td>
<td>an iterable object (riter) on (r)</td>
</tr>
<tr>
<td>next(riter)</td>
<td>the next random Color object from (riter)</td>
</tr>
</tbody>
</table>

Problem 4. In the test client (_main()) in randomcolors.py, we sorted the list colors containing \(n\) random Color objects in the order of their luminosities (see definition _cmp_(self, other)). How would you rewrite the statement colors.sort() to
a. sort the list in the order of the blue components of the colors?

b. sort the list in the order of the distance of the colors from black, i.e., \((0, 0, 0)\)? If we have a color \(c = (r, g, b)\), we define its distance from black as \(r + g + b\).

**Solutions**

**Solution 1.**

```python
# circle.py
import math

class Circle:
    def __init__(self, h=0.0, k=0.0, r=1.0):
        self._x = h - r
        self._y = k - r
        self._s = 2 * r

    def area(self):
        r = self._s / 2
        return math.pi * r ** 2

    def contains(self, x, y):
        r = self._s / 2
        h = self._x + r
        k = self._y + r
        return (x - h) ** 2 + (y - k) ** 2 <= r ** 2

    def __lt__(self, other):
        return self.area() < other.area()

    def __eq__(self, other):
        return self._x == other._x and self._y == other._y and \
               self._s == other._s

    def __str__(self):
        r = self._s / 2
        h = self._x + r
        k = self._y + r
        return '(' + str(h) + ', ' + str(k) + ', ' + str(r) + ')

def _main():
    import stdio
    c1 = Circle(1.0, 1.0, 2.0)
    c2 = Circle()
    stdio.writeln(c1.area())
    stdio.writeln(c1.contains(1.2, 2.2))
    stdio.writeln(c1 < c2)
    stdio.writeln(c1 == Circle(r=2.0, h=1.0, k=1.0))
    stdio.writeln(c1)

if __name__ == '__main__':
    _main()
```

**Solution 2.**

```python
# color.py

class Color:
    def __init__(self, r=0, g=0, b=0):
        self._r = r
        self._g = g
        self._b = b

    def getRed(self):
        return self._r
```
def getGreen(self):
    return self._g

def getBlue(self):
    return self._b

def luminosity(self):
    return (.299 * self._r) + (.587 * self._g) + (.114 * self._b)

def __add__(self, other):
    r = (self._r + other._r) // 2
    g = (self._g + other._g) // 2
    b = (self._b + other._b) // 2
    return Color(r, g, b)

def __eq__(self, other):
    return self._r == other._r and self._g == other._g and self._b == other._b

def __cmp__(self, other):
    if self.luminosity() < other.luminosity():
        return -1
    elif self.luminosity() == other.luminosity():
        return 0
    else:
        return 1

def __str__(self):
    return '(' + str(self._r) + ', ' + str(self._g) + ', ' + str(self._b) + ')

_class__main__()
import stdio

c1 = Color(23, 45, 156)
c2 = Color(34, 101, 78)
c3 = c1 + c2
a = [c1, c2, c3]
a.sort()
for v in a:
    stdio.writeln(str(v) + ' ' + str(v.luminosity()))
stdio.writeln(c1 == c2)
stdio.writeln(c1 == Color(23, 45, 156))

if __name__ == '__main__':
    __main__()

Solution 3.

# randomcolors.py

import stdrandom
from color import Color

class RandomColors:
    def __init__(self, n):
        self._n = n
        self._current = 0

def __iter__(self):
    return self

def next(self):
    if self._current >= self._n:
        raise StopIteration
    self._current += 1
    r = stdrandom.uniformInt(0, 256)
    g = stdrandom.uniformInt(0, 256)
    b = stdrandom.uniformInt(0, 256)
    return Color(r, g, b)
b = stdrandom.uniformInt(0, 256)
return Color(r, g, b)

def _main():
    import stdio
    import sys
    n = int(sys.argv[1])
    colors = []
    for color in RandomColors(n):
        colors += [color]
    colors.sort()
    for color in colors:
        stdio.writeln(color)

if __name__ == '__main__':
    _main()

Solution 4.

a. colors.sort(cmp = lambda x, y: cmp(x.getBlue(), y.getBlue()))
b. colors.sort(cmp = lambda x, y: cmp(x.getRed() + x.getGreen() + x.getBlue(), y.getRed() + y.getGreen() + y.getBlue()))

15 Case Study: N-body Problem

Problem 1. Consider three bodies $a$, $b$, and $c$ with masses $m_a = 1$, $m_b = 2$, and $m_c = 1$, initial positions $r_a = (-1,0)$, $r_b = (0,0)$, and $r_c = (1,0)$, and initial velocities $v_a = (0,1)$, $v_b = (0,0)$, and $v_c = (0,-1)$.

a. What are the forces $f_a$, $f_b$, and $f_c$ on each of the three bodies due to the other two? You may assume $G = 1$.
b. What are the velocities $v_a$, $v_b$, and $v_c$ of the three bodies after a second?
c. What are the positions $r_a$, $r_b$, and $r_c$ of the three bodies after a second?
d. What is the Python code for constructing objects representing bodies $a$, $b$, and $c$?
e. What is the Python code for calculating the forces $f_a$, $f_b$, and $f_c$ on the three bodies?
f. What is the Python code for updating the positions and velocities of the three bodies after a second?

Solution 1.

a. The force on body $a$ due to object $b$, denoted $f_{ba}$, is calculated as $f_{ba} = \frac{m_am_b}{|r_a-r_b|^2}(r_a - r_b)$. So, $f_a = f_{ba} + f_{ca} = (2.25,0)$, $f_b = f_{ab} + f_{bc} = (0,0)$, and $f_c = f_{ac} + f_{bc} = (-2.25,0)$.
b. The velocity of body $a$ after a second, denoted $v_a$, is calculated as $v_a = v_a + \frac{a}{m_a}f_1$. So, $v_a = (2.25,1)$, $v_b = (0,0)$, and $v_c = (-2.25,-1)$.
c. The position of body $a$ after a second, denoted $r_a$, is calculated as $r_a = r_a + v_a$. So, $r_a = (1.25,1)$, $r_b = (0,0)$, and $r_c = (-1.25,-1)$.
d. $a = Body(Vector([-1.0, 0.0]), Vector([0.0, 1.0]), 1.0)$
   $b = Body(Vector([0.0, 0.0]), Vector([0.0, 0.0]), 2.0)$
   $c = Body(Vector([1.0, 0.0]), Vector([0.0, -1.0]), 1.0)$
e. $fa = a.forceFrom(b) + a.forceFrom(c)$
   $fb = b.forceFrom(a) + b.forceFrom(c)$
   $fc = c.forceFrom(a) + c.forceFrom(b)$
f. $a.move(fa, 1)$
   $b.move(fb, 1)$
   $c.move(fc, 1)$
16 Searching and Sorting

Problems

Problem 1. Which order-of-growth class does the following function \( f() \) belong to, where the argument \( a \) is a list of \( N \) floats?

```python
def f(a):
    return sum(a) / len(a)
```

Problem 2. Suppose the running time of an algorithm on inputs of size 1000, 2000, 3000, 4000, and 5000 is 5 seconds, 20 seconds, 45 seconds, 80 seconds, and 125 seconds, respectively. Which order-of-growth class does the algorithm belong to?

Problem 3. How much memory (in bytes) does the list `['Alice', 'Bob', 'Carol']` occupy, assuming that a string of \( n \) characters occupies 37 + \( n \) bytes and a list of \( n \) items occupies 72 + 8\( n \) bytes?

Problem 4. Suppose we are searching each of 1000 keys in a sorted list of 8192 keys.
   a. How many comparisons are necessary in the worst case if we use linear search?
   b. How many comparisons are necessary in the worst case if we use binary search (use base-2 logarithm)?

Problem 5. Consider sorting an array \( a[] \) containing the following keys, by calling the `sort()` method (shown below) from insertion sort:

```python
   E A S Y Q U T I O N
```

```python
def sort(a):
    n = len(a)
    for i in range(1, n):
        j = i
        while (j > 0) and (a[j] < a[j - 1]):
            exchange(a, j - 1, j)
            j -= 1
```

What is \( a[] \) when \( i = 5 \) (ie, 5th iteration of the outer loop) and the inner loop (lines 5 - 7) is complete?

Problem 6. Which of the following sequences can be merged using the merge operation?
   a. HELLO and WORLD
   b. E H L L O and D L O R W

Solutions

Solution 1. Linear, ie, \( T(N) = N \)

Solution 2. Quadratic, ie, \( T(N) = N^2 \)

Solution 3. 220 bytes

Solution 4.
   a. \( 1000 \times 8192 = 8.192 \times 10^6 \)
   b. \( 1000 \times \log 8192 = 1.3 \times 10^4 \)

Solution 5. AEQSUYTION

Solution 6.
   a. No.
   b. Yes.
17 Stacks and Queues

Problems

Problem 1. Suppose that a minus sign in the input indicates pop the stack and write the return value to standard output, and any other string indicates push the string onto the stack. Further suppose that following input is processed:

    it was - the best - of times - - - it was - the - - worst - of times -

a. What is written to standard output?

b. What are the contents (top to bottom) left on the stack?

Problem 2. Consider the following code fragment:

```python
s = Stack();
while n > 0:
    s.push(n % 2)
    n = n / 2
while not s.isEmpty():
    stdio.write(s.pop())
stdio.writeln()
```

a. What does the following code fragment print when `n` is 50?

b. Give a high-level description of what it does when presented with a positive integer `n`.

Problem 3. Suppose that a minus sign in the input indicates dequeue the queue and write the return value to standard output, and any other string indicates enqueue the string onto the queue. Further suppose that following input is processed:

    it was - the best - of times - - - it was - the - - worst - of times -

a. What is written to standard output?

b. What are the contents (head to tail) left on the queue?

Problem 4. What does the following code fragment do to the queue `q`?

```python
s = Stack()
while not q.isEmpty():
    s.push(q.dequeue())
while not s.isEmpty():
    q.enqueue(s.pop())
```

Solutions

Solution 1.

a. was best times of the was the it worst times

b. of it

Solution 2.

a. 110010

b. The binary representation of `n`

Solution 3.

a. it was the best of times it was the worst

b. of times

Solution 4. Reverses the items on the queue.
18 Case Study: Small-world Problem

Problems

Problem 1. Consider an undirected graph \( G \) represented by the following file `tinygraph.txt`, and let \( g \) be a `Graph` object built from the file:

```
1 2
1 4
2 3
2 4
2 5
3 5
4 5
```

a. What does the adjacency-list representation of \( G \) (ie, the dictionary \( g._adj \)) look like?

b. How many vertices and edges does \( G \) have, ie, what are the values of \( g.countV() \) and \( g.countE() \)?

c. What is the degree of vertex 2 and vertex 4, ie, what are the values of \( g.degree(2) \) and \( g.degree(4) \)?

d. If \( pf \) is a `PathFinder` object constructed as \( pf = PathFinder(g, 1) \), what do the dictionaries \( pf._distTo \) and \( pf._edgeTo \) look like? What is the shortest distance and shortest path from vertex 1 to vertex 5?

e. What is the average degree of \( G \), ie, what is the value of \( smallworld.averageDegree(g) \)?

f. What is the average path length of \( G \), ie, what is the value of \( smallworld.averagePathLength(g) \)?

g. What is the clustering coefficient of \( G \), ie, what is the value of \( smallworld.clusteringCoefficient(g) \)?

Solutions

Solution 1.

```
a. 1  2  4
    2  1  3  4  5
    3  2  5
    4  1  2  5
    5  2  3  4
```

b. 5 vertices and 7 edges

c. 4 and 3

d. \( pf.distTo = \{1:0, 2:1, 3:2, 4:1, 5:2\} \) \( pf.edgeTo = \{1:None, 2:1, 3:2, 4:1, 5:2\} \); shortest distance from 1 to 5 is 2 and the shortest path from 1 to 5 is 1-2-5.

e. 2.8; if \( V \) and \( E \) denote the number of vertices and edges of \( G \), then the average degree of \( G \) is calculated as \( \frac{2E}{V} \).

f. 1.3; the average path length of \( G \) is the sum of shortest path lengths between all pairs of vertices divided by the number of such pairs, which is \( V(V-1) \).

g. 0.767; if \( d \) is the degree of vertex \( v \), ie, the number of neighbors of \( v \), the clustering coefficient of \( v \) is number of those neighbors that are actually connected divided by the number of possible connections among them, which is \( d(d-1) \); the clustering coefficient of \( G \) is the average value of the clustering coefficient over all the vertices of \( G \), ie, the sum of the clustering coefficient of each vertex of \( G \) divided by the number of vertices \( V \).
19  Limits of Computation

PROBLEMS

Problem 1. Classify each of the following programs/problems as “never ending”, “easy”, “hard”, or “impossible”.

a. # mystery .py

    def f():
        i = 1
        while i > 0:
            stdio.writeln(i)
            i += 1

        f()

b. Multiplying two \( N \times N \) matrices.

c. Computing an optimal tour for a traveling salesperson, ie, a tour of lowest total cost that starts in some (home) city, includes each city exactly once, and ends at the home city.

d. A completely reliable virus detection program.

Problem 2. Classify each of the following sets as “finite”, “countably infinite”, or “uncountably infinite”.

a. The set of integers from the interval \([10, 100]\).

b. The set of multiples of 3 from the interval \([0, \infty]\).

c. The set of real numbers from the interval \([0, 0.1]\).

Solutions

Solution 1.

a. Never ending

b. Easy; can be solved in cubic \( T(N) = N^3 \) time

c. Hard; involves searching through \( N! \) tours if there are \( N \) cities

d. Impossible; equivalent to solving the halting problem, which is impossible

Solution 2.

a. Finite

b. Countably infinite

c. Uncountably infinite