Computational Management Science 1

Fall 2021 Final

registration number: .....

(Do not write your name on the test - just the 7 digit student id number.)

All examples are evaluated using the Python programming language, version 3.8.

1. (6 points) Writing Code

   (a) (3 points, ≤5 minutes) Functions
   Write a function `fib(n)` that takes exactly one non-negative integer as argument and returns the n-th fibonacci number. Use either a loop or recursion to compute the solution. Provide a proper docstring and a doctest. [Note: a fibonacci number is the sum of it's two predecessors; the first numbers are 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...]. Hence, for example
   `fib(0) = 0, fib(1) = 1, fib(2) = 0 + 1 = 1, fib(3) = 1 + 1 = 2, ....`

```
def fib(n: int) -> int:
    """
    Return the nth fibonacci number.

    >>> fib(3)
    2
    """
    return n if n <= 1 else fib(n - 1) + fib(n - 2)
```

   (b) (3 points, ≤5 minutes) Functional programming
   Use a lambda function and the correct higher order function to generate the squares of a given sequence `seq` of integral numbers. Alternatively, you can use list comprehensions. Store the result in a new variable. For example `seq = [1, 4, 5, 5, 3, 7]` yields `[1, 16, 25, 25, 9, 49]`.

```
result = map(lambda x: x*x, seq)
```

2. (6 points + 2 bonus, ≤10 minutes) Relational Database Design

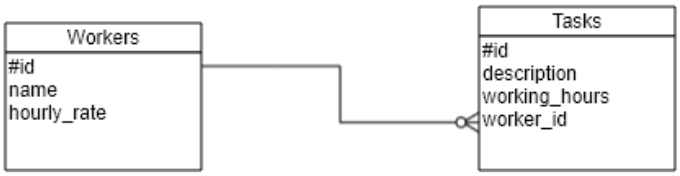
Design a relational database to store the following information:

We want to store a list of workers for which we know the "name" and the "hourly rate" in Euros per hour. In addition, we want to store a list of tasks for which we know a "description", number of "working hours needed" (integer) and "location" (string).

(a) Every task is assigned to a unique worker which performs the task.
In addition, please also give an example on how to store the case of 2 workers and 3 tasks, the first two being assigned to the first worker, the third assigend to the second worker. Assign arbitrary but different "hourly rate" and "working hours needed" to the workers and tasks.
For your example, compute the total cost of the given assignment.

(b) Every task can be assigned on none, one or multiple workers and the number of "assigned hours" of a worker to a task should also be stored.
In addition, please also give an example with two workers and two tasks. The first task should be assigned to the 2nd worker and the 2nd task should be assigned to both workers. Assign arbitrary but different "hourly rate" and "working hours needed" to the workers and tasks.
For your example, ensure that enough working hours are assigned to each job and compute the cost of the assignment.

(c) **Bonus**: Give an SQL query that compute the total cost of the stored assignments for case (a).



(a)

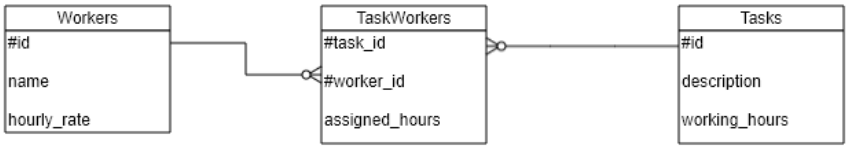
Workers

id	name	hourly_rate
1	Sabine	100
2	Thomas	80

Jobs

id	description	working_hours	worker_id
1	loops-y	42	1
2	recursion-b	250	1
3	sums-b	10	2

Total cost: $42 \cdot 100 + 250 \cdot 100 + 10 \cdot 80 = 30000$



(b)

Workers

id	name	speed
1	Sabine	100
2	Thomas	80

Tasks

id	description	working_hours
1	loops-y	42
2	recursion-b	250

TaskWorkers

task_id	woker_id	assigned_hours
1	2	42
2	1	150
2	2	100

Total cost: $42 \cdot 80 + 150 \cdot 100 + 100 \cdot 80 = 26360$

(c)

```
SELECT SUM(jobs.working_hours * workers.hourly_rate)
FROM jobs
INNER JOIN woerkers ON jobs.worker_id=workers.id;
```

3. (9 points, ≤10 minutes)

   (a) (1 points)
   Name a downside of the Python programming language.
   Execution speed.

   (b) (3 points)
   Name at least three characteristics, concepts or benefits of functional programming.

   i. Functions are first class citizens.
   ii. Recursion.
   iii. Higher order functions.

   (c) (3 points)

   - What is the asymptotic complexity of calculating the sum of all elements in a list with n elements?
     n
   - What is the asymptotic complexity of sorting a list of numbers?
     $n \log(n)$
   - Simplify this term: $O(3n + 5n^2)$
     $O(n^2)$

   (d) Finally, name one thing you like about this course and one thing that should be improved in the future (be honest!) (2p).
   individual responses

4. (12 points, ≤10 minutes) Reading and Understanding Code
What is the output of the following code snippets? Write exactly what the output of each snippet is if the snippet is the sole content of a Python file. If the output is an error message, it is enough to write "ERROR". If there is no output, write "-".

```
if __name__ == "__main__":
    import sys
    print(sys.exit(0))
```

```
-
```

```
def decode(text):
    return encode(text)
print(decode('apt-build moo'))
```

```
  ERROR
```

```
from collections import namedtuple
Point = namedtuple('Point', ('x', 'y'))
A = Point(x=2, y=3)
```

```
-
```

```
sum = lambda x, y: x + y
print(sum(3, 4))
```

```
7
```

```
s = "Everyone wants to become a great programmer."
print(s[9:13], s[-11:-1], end='')
print(s[13])
```

```
want programmers
```

```
integers = (i**2 for i in range(10) if i % 3 == 0)
print(max(integers))
```

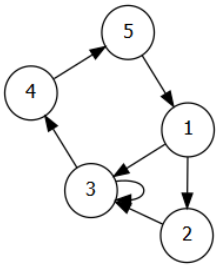
```
81
```

5. (9 points, ≤10 minutes)

   (a) (3 points)
   What is recursion? What alternative programming concept can you always use instead of recursion? Write a short code example of a recursive function.
   A function calling itself.
   You can always use loops instead of recursion.

```
def binary_search(sorted_list, l, r, key):
    if r >= l:
        m = (l + r) // 2
        if sorted_list[m] == key:
            return m
        elif sorted_list[m] > key:
            return binary_search(sorted_list, l, m-1, key)
        elif
            return binary_search(sorted_list, m+1, r, key)
    else:
        return -1
```

   (b) (3 points)
   What are common ways to represent a graph (give two such represenetations)? Draw an example of a graph or digraph (with at least 4 vertices and 5 edges/arcs) and how you would store it in two of those representations! Name at least one advantage of each of your two representations!
   Adjacency matrix, adjacency list



   Adjacency matrix:

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

   Adjacency list:

```
    graph = {1: [2, 3],
             2: [3],
             3: [3, 4],
             4: [5],
             5: [1]}
```

   Advantages: adjacency matrix: Testing adjacency is $O(1)$; adjacency list: requires less memory for sparse graphs.

(c) (3 points)
What is a game with perfect information game? In addition to a general explaination, give one example of a game with perfect information and one example of a game without perfect information.

All player know everything about the games state

Games without perfect information: Poker, Bridge

Games with perfect information: Chess, Go

Unclear case (depend on exact definition) due to chance events: Backgammon

6. (6 points, ≤5 minutes) Student Points
A fictitious lecturer has to keep track of the points of their students. At the end of the course they want to compute the total number of points for each student. The points given to the students for each activity are stored in a list of tuples, where each tuple has three entries, the first being the id of the student as a string, the a string giving the reason for awarding the points and the third the number of points awarded to the student. In order to know the total number of points per student, this list should be transformed into a dictionary with student id's as keys and the sum of points awarded to a student as values. The sort order of students and the reasons for awarding the points not relevant. Also students that did not receive any points are not relevant, since they will not be graded. Write a function `compute_total_points(point_assignments)` that returns a new dictionary in the desired format. For example

```
point_assignments = [('9825756', 'hw_1', 5),
    ('0315423', 'hw_1', 4), ('1303451', 'bonus', 2),
    ('0315423', 'hw_2', 6), ('1303451', 'hw_2', 5),
    ('0315423', 'bonus', 1)]

total_points = compute_total_points(point_assignments)
```

Then the variable `total_points` will store the same dictinary as given by:

```
total_points = {'9825756': 5, '0315423': 11, '1303451': 7}
```

```python
def compute_total_points(point_assignments):
    total_points = {}
    for id, _, points in point_assignments:
        if id not in total_points:
            total_points[id] = points
        else:
            total_points[id] += points
    return total_points

def compute_total_points(point_assignments):
    return {id: sum(assignment[2]
        for assignment in point_assignments if assignment[0] == id)
        for id in set(point_assignments.keys())}
```