Due: 4th of May 2025 at 11:59pm

Submit your assignment via Canvas (Q1 and Q2) and Ed (Q3). Submission for Q1 and Q2 must be in pdf format and cannot be handwritten. This assignment is worth 15% of your final grade.

COMP9123 Data Structures and Algorithms – Assignment 2

Semester 1 2025

As a first step go to the last page and read the section: Advice on how to do the assignment. Note that providing code is optional for Q1 and Q2, but explanations are mandatory.

Question 1

In the lectures, we saw how to construct a priority queue using heaps and lists. It is also possible to construct a priority queue using an AVL tree.

Describe how to perform the three main operations of a priority queue using an AVL tree as the underlying data structure. The operations `insert`(k, v) and `remove-min`$()$ should take $O(\log n)$ time, and `min`$()$ should run in $O(1)$ time.

For simplicity, you can assume that the keys will all be unique.

a) Design a priority queue using an AVL tree as the underlying data structure. **(5 marks)**

b) Argue the correctness of your operations. **(2 marks)**

c) Analyse the running time of your operations. **(3 marks)**

Question 2

Bob Proverra decides to expand his small cafe into a restaurant, and as a new promotion, he has decided to advertise how many lunch specials he has that cost at most \$X. Every lunch special consists of one main dish and one side dish. His chefs like to try out new dishes regularly, so it would be great if he could efficiently compute the number of such combinations when needed.

Your task is to design a data structure that supports the following operations, where `name` is the name of a dish and `price` is its price (a positive integer):

- `addNewMainDish(name, price)`: Adds a new main dish called `name` costing `price`.
- `addNewSideDish(name, price)`: Adds a new side dish called `name` costing `price`.
- `removeMainDish(name)`: Removes the main dish called `name` and returns its `price`, if `name` exists.
- `removeSideDish(name)`: Removes the side dish called `name` and returns its `price`, if `name` exists.
- `countCombinations()`: Returns the number of combinations of a main dish and a side dish with a total price at most \$X.

Each operation should run in $O(1)$ expected time, and the data structure should take $O(n)$ space. Note that X is not a constant and the time or space complexity should not depend on it.

a) Design a data structure that supports the above operations in the required time.

(6 marks)

b) Argue the correctness of your data structure and its operations. **(2 marks)**

c) Analyse the running time of your operations and the total space complexity of the data structure. **(2 marks)**

Question 3

Quadtrees are a type of tree which has exactly 4 children. Your task is to implement a Quadtree representation of images. Our representation of quadtrees will not have a separate `Node` class. Instead, a `QuadtreeImage` itself can be thought of as a node. A `QuadtreeImage` is either:

- A **leaf**, representing a square region of pixels all of the same colour.
- An **internal node**, containing four child `QuadtreeImage` objects, one per quadrant of the bitmap image.

You must implement your submission in Python. The only file which you need to edit is `quadtree_image.py`, which contains the class `QuadtreeImage` and the methods you need to implement:

- `blackenNorthWestQuadrant()` : Blacken the entire north-west quadrant.
- `countPixels(Colour)` : Count pixels of a given colour in the image represented by the quadtree.
- `invertColours()` : Invert the colours in the image represented by the quadtree, i.e., turn every black pixel white and every white pixel black.
- `setPixel(int x, int y, Colour)` : Change the colour of a single pixel in the image represented by the quadtree to the specified colour.
- `computeOverlay(QuadtreeImage bmp1, QuadtreeImage bmp2)` : Construct and return the overlay of the two input images of the same size. In the overlay, a pixel is black if either of the input images has a black pixel in the same location. That is, a pixel in the output image is white only when the corresponding pixel in both input images is white; otherwise, the output pixel is black. Rather than perform the operation pixel by pixel, one can compute the overlay more efficiently by leveraging the quadtree's ability to represent multiple pixels with a single node.

Note: For full marks, the resulting quadtree must be in *reduced form*, i.e., any subtree that is entirely one colour must be reduced to a single leaf.

a) Implement your data structure and algorithms on Edstem.
`https://edstem.org/au/courses/20134/lessons/78478/slides/529336`

(10 marks)

Implementation Details

Empty method definitions have been provided in the `QuadtreeImage` class for each of the methods you need to implement. They start from Line 90 of the Python scaffold.

We have provided a scaffold which includes many helpful utilities, such as:

Helper Methods

- A method that constructs a `QuadtreeImage` from a string representation (see an example of the expected format at the end).
- A method that converts a `QuadtreeImage` into a string representation.

Constructors

- A constructor that creates a leaf `QuadtreeImage` for a given location and size.
- A constructor that creates a `QuadtreeImage` for a given location and size, with quadrants specified by a list of `QuadtreeImage` objects.

Convenience Methods

- A method that enumerates quadrants in the following order: **northeast, northwest, southeast, southwest**.
- A method that determines which quadrant an input (x, y) coordinate pair lies within (if any).
- A convenience method that reads a string representation of an image from standard input and constructs the corresponding `QuadtreeImage`.
- Methods that generate string representations of a `QuadtreeImage`, both as a direct representation of the image it encodes and with boxing interspersed to depict its tree structure.

Testing and Experimentation

A `main` method, which will not be tested, has also been provided. You may use it for experimentation and testing. It is located at the top of the Python scaffold.
If you run the scaffold code in its initial state, you will be prompted to enter a string representing an image. Try entering the following example:

```
........
.....***
...***..
..**....
..*.....
.**...**
**...***
*....***
```

Important Notes

We advise against modifying the other methods provided in the scaffold. However, you are allowed to:

- Write additional methods, classes, etc.
- Add additional source files to your assignment workspace as needed.

Please ensure that you comply with the University's plagiarism policy when writing your code.

Advice on how to do the assignment

- All submitted work must be done individually without consulting someone else's solutions in accordance with the University's "Academic Dishonesty and Plagiarism" policies.

- Assignments submitted via Canvas as pdf (no handwriting!)

- Be careful with giving multiple or alternative answers. If you give multiple answers, then we will give you marks only for ”your worst answer”, as this indicates how well you understood the question.

- Some of the questions are very easy (with the help of the lecture notes or book). You can use the material presented in the lecture or book (without proving it). You do not need to write more than necessary (see comment above).

- When giving answers to questions, we always would like you to prove/explain/motivate your answers.

- When giving an algorithm as an answer, the algorithm does not have to be given as (pseudo-)code.

- If you do give (pseudo-)code, then you also have to explain your code and your ideas.

- Unless otherwise stated, we always ask about worst-case analysis, worst case running times etc. (for example, hashing does not guarantee anything in the worst case, in general).

- As done in the lecture, and as it is typical for an algorithms course, we are interested in the most efficient algorithms.

- It might help you (and us) if you briefly describe your general idea, and after that you might want to develop and elaborate on details. If we don't see/understand your general idea, we cannot give you points for it.

- If you use further resources (books, scientific papers, the internet,...) to formulate your answers, then add references to your sources.

- If you refer to a result in a scientific paper or on the web you need to explain the results to show that you understand the results, and how it was proven.