

Assignment #1 - Part [B]

Task 1: Fibonacci Series [6 Marks]

Define functions to compute the nth Fibonacci number using the below three methods.

- Implement the Fibonacci series using recursion. **[2 marks]**
- Implement the Fibonacci series using divide and conquer (matrix multiplication) **[2 marks]**
- Implement the Fibonacci series using dynamic programming. **[2 marks]**

Task 2: Search Algorithms [8 Marks]

Implement sequential and binary search algorithms, demonstrating both iterative and recursive approaches.

- Sequential Search: **[2 marks]**
 - Define a function that takes a list and a target value as input.
 - Implement the sequential search iteratively.
 - If the target value is found, return the index, otherwise, return an indication that the target is not found.
- Recursive Sequential Search: **[2 marks]**
 - Define a recursive function that takes a list and a target value as input.
 - Search recursively for the target value, if found, return the index, otherwise, return an indication that the target is not found.
- Binary Search **[2 marks]**
 - Define a function that takes a sorted list and a target value as input.
 - Implement the binary search iteratively.
 - If the target value is found, return the index, otherwise, return an indication that the target is not found.
- Recursive Binary Search **[2 marks]**
 - Define a recursive function that takes a list, a target value, low and high index as input.
 - Search recursively for the target value, if found, return the index, otherwise, return an indication that the target is not found.

Task 3: Heap, Priority Queue and Heap Sort [6 Marks]

1. Implement a Heap: [2 marks]
 - a. Define a class/structure for the heap.
 - b. Implement methods for inserting an element and maintaining the heap property (heapify).
 - c. Implement methods to extract the maximum and minimum from the heap.
2. Implement a Priority Queue: [2 marks]
 - a. Use the heap implementation (from step 1) to build a priority queue.
 - b. Define methods for inserting an element with a priority and extracting the highest priority element.
 - c. Ensure the insertion maintains the heap property.
3. Implement Heap Sort: [2 marks]
 - a. Utilize your heap implementation to sort an array.
 - b. Build a max heap from the array and repeatedly extract the max element from the heap and rebuild it until the array is sorted.

Submission Guidelines:

- You should work in teams of 3
- Teams can be of different groups/major
- No late submissions are allowed.
- Cheating is NOT tolerated by any means.
- TAs will grade the assignment out of 20, but this score may be adjusted later for scaling purposes.

Deliverables:

- ONLY the team leader should submit a Zip file under the name:
<G#_TeamLeaderName_TeamLeaderID>
- The zip file should include:
 - A text file with the teams' names and IDs
 - A CPP program for the three Fibonacci functions.
 - A CPP program for the four Searching methods.
 - A CPP program for the heap, priority queue and heap sort.