UENG-101 (2025): Practice Midterm Max. Marks: 50. Time: 2 hours.

Instructions to students

IMPORTANT**:** Read these **now**. We will **not** print these on the question paper.

1. Students must report 10 mins prior to the commencement of the exam & sit in their assigned seats only.
2. Students are not allowed to leave the examination hall within 1 hr after the commencement of the exam.
3. Before answering the questions, the student must write his/her name and student registration number on the answer script.
4. Mobile phones/any form of communication devices are strictly prohibited in the exam rooms. It is best to leave the mobile phone in the hostel before coming for the exam.
5. Mere possession of a mobile phone/communication device inside the examination hall or during the exam will be treated as a case of cheating (absolutely no excuses). In such cases, suitable action will be taken.
6. **With the exception of one A4-sized cheat sheet** (double-sided, hand-written or typed in any font size), no form of unfair means (including talking to another student, copying from another student's paper, copying from any books, notes, etc., entering the examination hall with writing on one's clothing or person, use of mobile phone/ communication devices) during the examination will be tolerated. A student found to have resorted to any form of unfair means during the examination will be given an 'F' grade in that course as a minimum punishment. No appeals will be accepted.
7. Abetting a student to resort to any form of unfair means will also be considered an unfair practice. In this instance as well, the student abetting another student will be given an 'F' grade in that course. No appeals will be accepted. In case the student who is abetting another student is from a different class/batch, suitable action will be taken against such a student.
8. When an additional sheet is taken by the student, the student must write their name and serial number, sign the additional sheet and must get it countersigned by the invigilator.
9. **No calculators** will be allowed during the examinations.
10. If a student is found with a mobile phone/communication device while taking a break to use the washroom when the exam is in progress, it will be treated as a case of cheating. Irrespective of whether the student is using the mobile phone/communication device or not, the same penalty as in item 5 will be applicable.

1. [**2 + 2 + 2 marks**] Suppose queues Q_1 and Q_2 are initially empty. We first **enqueue** each value 1, 2, 3, 4, and 5 (in that order) into either Q_1 or Q_2. We then perform a sequence of **dequeue** operations (on either Q_1 or Q_2, as long as that queue is non-empty). Write a sequence of **enqueue** and **dequeue** operations so that the order in which values are dequeued precisely matches each sequence below, or show why no such sequence of enqueues and dequeues is possible.

   a) 3, 4, 5, 1, 2
   b) 3, 4, 5, 2, 1
   c) 1, 3, 4, 5, 2

2. [**2 + 2 marks**] We are comparing two implementations A and B of an algorithm. On any input of size n, the worst-case asymptotic running times of implementations A and B are $O(n^2)$ and $O(n^2 \log n)$ respectively.

   a) Clearly describe when you would recommend using implementation A.

   b) Clearly describe when you would recommend using implementation B.

3. [**10 + 3 marks**] This `mystery` function makes use of a recursive `helper` function (line numbers shown for clarity).

```
 1: def helper(s: str, lo: int, hi: int) -> bool:
 2:     if lo == len(s):
 3:         return False
 4:     if lo == hi - 1:
 5:         return helper(s, lo + 1, len(s))
 6:     if s[lo].lower() == s[hi - 1].lower():
 7:         return True
 8:     else:
 9:         return helper(s, lo, hi - 1)

10: def mystery(s: str) -> bool:
11:     return helper(s, 0, len(s))
```

   a) Use the given line numbers to **precisely** trace the execution for the following function call: `mystery("Momo")`. Use the following tabular format:

Line number	Values of relevant variables	Remarks
10	`s = "Momo"`	Parameter initialized

   b) Suggest a helpful function name and a detailed docstring for the `mystery` function.

4. [**3 + 4 + 4 marks**] Starting at the root node of a binary search tree (BST), we observe this sequence of nodes while searching for a key with value k: 2, 252, 401, 398, 344, 397, 363.
   a) Draw a BST consistent with the above sequence.

   b) Since the sequence terminates with 363, your friend claims that $k = 363$. To show that your friend *may* be incorrect, identify *all* values of k consistent with this sequence.

   c) Your friend wants to write a Python function to determine all possible values of k given such a sequence. Your friend suggests the following signature:

   ```
   def findkey(sequence: list[int]) -> int:
   ```

   Improve the above signature in ***at least*** three ways.

5. [**4 + 8 + 4 marks**] Recall the Python implementation of linked lists from our Lab. Given a linked list, your friend wants to write a function `remove_ints` that returns a Python list containing only the non-`int` objects in the linked list. However, the relative order of any two non-`int` objects should be *reversed* in the resulting list.

   a) Demonstrate that you have understood what your friend wants by completing these doctests:
      i. `>>> remove_ints([1, [0.0, None]])`
      ii. `>>> remove_ints([None, ['None', [0, [[], None]]]])`

   b) Your friend plans to implement the Python function as follows:
      i. Initialize an empty linked list `temp = None`
      ii. Process each value in the given linked list (left to right) and *prepend* it to the `temp` linked list.
      iii. After processing all values, initialize an empty list `result = []`
      iv. Process each value in the `temp` linked list (left to right). If the value is not of type `int`, append it to the `result` list.
      v. After processing all values, return the `result` list.

   Demonstrate your understanding of the above algorithm by specifying the `result` list for each of the given doctests. Clearly explain how the `result` is calculated and also indicate whether any of these doctests **fail** (indicating that the algorithm is buggy).

   c) Your friend claims that the worst-case asymptotic running time of the `remove_ints` function can be improved by prepending each value to the `temp` list and returning `temp` at the end of step (iii) after all processing all values. Clearly explain whether your friend is correct or incorrect.
   For more practice: Your friend claims that the worst-case asymptotic running time will improve if we prepend just the non-`int` values to `temp` in step (iii), rather than delaying this to step (iv). Clearly explain whether your friend is correct or incorrect.