CP 312, Winter 2023
Assignment 4 (7.5% of the final grade)
(due Monday, March 26, at 23:30)

There are five questions in this assignment.

1. [6 marks] **Greedy: Making the deadline.** In the **Deadline** problem, an instance is a set of n tasks such that each takes 1 (one) unit of time to complete and a set of deadlines $d_1, d_2, \ldots, d_n$. When a task is not completed by its deadline, a fixed penalty of 100$ is applied. A valid solution to the **Deadline** problem is an ordering of the tasks that minimizes the total penalty incurred when the tasks are completed one after another in the order provided starting at time 0.
   For example, if the instance is $n = 4$ and $d_1 = 2, d_2 = 4, d_3 = 5, d_4 = 2$ then the order of execution of tasks $1, 4, 2, 3$ will give the minimal penalty of 0.
   Another example, if the instance is $n = 4$ and $d_1 = 2, d_2 = 1, d_3 = 4, d_4 = 2$ then the order of execution of tasks $2, 1, 3, 4$ will give the minimal penalty of 100.

   Design a greedy algorithm that solves the **Deadline** problem (i.e., takes an instance of this problem and prints the order of tasks that minimizes the total penalty, and also the corresponding penalty). Justify its correctness and analyze the running time.

2. [12 marks] **Dynamic programming**

   In the MinWeightChange problem, for a given list of positive integers $1 = d_1 < d_2 < d_3 < \ldots < d_n$ that represent the denominations of the n types of coins, a list of positive integers $w_1, \ldots, w_n$, where w_i is the weight of the coin of denomination d_i, and a target value V, we must find the weight of a minimum-weight set of coins (allowing multiple coins of a given denomination, as usual) whose denominations sum up to V.

   (a) Give an input showing that the minimum-weight set of coins with total value V is *not* always the set with the fewest number of coins with total value V.

   (b) A possible greedy algorithm for the problem considers the coins in the order defined by the ratio d_i/w_i in decreasing order. (As usual, the greedy algorithm adds as many of the first coin as possible without exceeding the target value, then adds as many of the second coin in that order, etc.) Give an example showing that this algorithm does not always find the optimal solution.

   Hint: You may want to consider inputs with denominations $d_1 = 1$, $d_2 = 2$, and $d_3 = 5$.

   (c) Give a precise definition of a subproblem that can be used to solve your problem with a dynamic programming algorithm.

   (d) Describe and justify the recurrence rule that will be used to compute the solutions to the subproblems defined in part (c).

you start with
"let weight[i] be the smallest weight of coins required to exchange the sum i"

and establish "similar" relation between weight[i] for different values of i
(this was A HINT).

(e) Provide a pseudocode for your dynamic programming algorithm that solves your problem using the subproblems and recurrence rule defined above. Your algorithm needs to find the minimal possible weight only (no need to find the set of coins).

(f) Analyze the time complexity of the algorithm obtained in part (e).

3. [6 marks] **Divide-and-Conquer/Binary Search.** Consider a sorted in non-descending order array of n integers in which all elements appear twice (one after one) and one element appears only once. Find that element. For example, if the given array is $[1, 1, 3, 3, 4, 5, 5, 7, 7, 8, 8]$, the output is 4, if the input array is $[1, 1, 3, 3, 4, 4, 5, 5, 7, 7, 8]$, the output is 8. Justify correctness and analyze the running time of your algorithm.
NOTE!!!: Your algorithm **MUST** have running time in $O(\log n)$, so simple scanning from left to right is not going to work.
Detailed pseudocode description is required for full mark in this question.

4. [20 marks] **Dynamic Programming and Reduce to Known; programming question**. A sequence of characters from the set $\{a, b, c, \ldots, z\}$ is a palindrome if it is equal to its own inverse. Examples: `abba`, `stopots`, etc. Given a string (sequence of characters) we would like to find its longest subsequence which is palindrome. For example, the string "strabetubsa" has the longest palindromic subsequence "abeba" (note that there are multiple palindromic subsequences of length 5 in this case, such as "abtba" or "stats").

   (a) [5 marks] Design Dynamic Programming algorithm to solve this problem. You can use reduce to known technique to reduce the longest palindromic subsequence problem to another problem that you know how to solve with DP approach. Justify its correctness, provide pseudocode and analyze the running time.
   Your solution must have worst case running time in $O(n^2)$ where n is the length of given string.

   (b) [15 marks] Write a Java or Python program that uses your algorithm to process multiple strings and find longest palindromic subsequence in each of them. You program will read multiple lines from the standard input and print the results to the standard output.

   Input Specification
   Input consists of several lines each containing one case:
   For example
   `strabetubsa`
   or
   `multiple`

   Observe that you can not assume any limit on the number of lines entered (empty line will signify the end of input data).

   Output Specification
   You program should print one longest palindromic subsequence for each line of input. You program **must not print any prompts** or anything else, just the output specified above!

Deliverables

You can use any IDE for your code development, but you always submit only single Java or Python file. If your Java solution requires multiple classes – use Java inner classes. Do not submit for example Eclipse projects, or any other meta-files. Single source file only! File name MUST be `"lpal.java"` or `"lpal.py"`.

If you use Java, make sure that this single file compiles outside of any IDE. To verify this, just place it in an empty temporary directory, open command line window, go to that directory and try to run

```
javac *.java
```

in command line window. If compiling is successful, try to run your program in command line window. For example,

```
java lpal
```

and verify that it works.

Sample input

```
this
is
simplest
sample
input
dsatatad
```

Sample output

```
t
i
sis
s
i
datatad
```

Note, that you only need to print one of possible longest subsequences, so there are multiple correct outputs in this example.

Submission.

1. Submit single pdf file `A4.pdf` and single lpal.java or lpal.py file to the assignment 4 dropbox on `MyLearningSpace`.
2. Do not submit any meta-files, archives or class-files!!!