CSC 2400 Spring 2025
Extra Credit Programming Project

Recursive Decrease-by-One and Conquer Algorithm: Finding the Maximum

🧠 Project Goal

In this extra credit programming project, you will implement and demonstrate a recursive algorithm that uses the decrease-by-one and conquer strategy to find the maximum element in an unsorted array. This project emphasizes not only correct implementation but also educational clarity — helping others understand this important algorithmic strategy.

🛠️ Your Task

1. Write a fully executable program in the language of your choice that:

- Uses a recursive, decrease-by-one and conquer approach.

- Accepts its input as parameters — do not hardcode the array.

2. Your program must clearly illustrate:

- The decrease phase: recursive calls decreasing the size of the array, moving toward the base case.

- The conquer phase: results combining during return from recursion.

3. You may produce:

- Static output (e.g., printed trace of recursive calls).

- Dynamic output (e.g., animations, delays, user interaction, audio, etc.).

🎯 The goal is to make your output clear, descriptive, and helpful for future students learning recursion and decrease-by-one and conquer techniques.

☑️ Required Input Example

Use the array S = [3, 5, 2, 9, 1, 8, 0, 2] for demonstration — but pass it as a parameter, not hardcoded in the function.

Example of Correct Algorithm in Pseudocode

Algorithm **RecursiveMaxFind**(S, n)

// Input: A sequence S with n elements

// Output: The maximum element in S

```
if n = 1 then
   return S[0]
else
   maxRest <- RecursiveMaxFind(S, n - 1)

   if maxRest > S[n - 1] then
      return maxRest
   else
      return S[n - 1]
```

Submission Requirements

- Source code file (must compile or run as-is)

- Sample output using the input S = [3, 5, 2, 9, 1, 8, 0, 2]

- Sample output can be a printout or a video demonstration

- A brief explanation (text or short video) that:

- Shows how your output illustrates the decrease and conquer phases

- Explains how this helps teach recursion

- Citations for all external sources used, including websites, textbooks, and generative AI

- Full names of all team members included in the code comments

Code Requirements

All code must begin with a top comment block that includes:

- Student name(s)

- How to run the code (including platform or command-line instructions)

- Required dependencies (if any)

- Whether the output is static or dynamic

- Where the educational output is located (e.g., console, animation, etc.)

🤖 Generative AI & External Source Policy

You may use generative AI tools or other external resources, but you must cite every tool or source used. This includes indicating whether AI was used for:

- Brainstorming ideas

- Writing code

- Debugging

- Designing or formatting output

- Writing your explanation

Refer to this official guide for citation instructions:

https://www.tntech.edu/citl/ai-citing.php

❌ No credit will be given for any submission that uses external sources without proper citation.

🤝 Team Policy

- Work may be done individually or with your homework team.

- Teams must use the same composition as your regular assignments.

- One submission per team.

- All member names must appear in the source code comments.

🔑 Grading Breakdown

☑ Correct recursive decrease-and-conquer algorithm — REQUIRED

☑ Output clearly shows decrease phase — REQUIRED

☑ Output clearly shows conquer phase — REQUIRED

☑ Code is executable and properly documented — REQUIRED

☑ Explanation of how the output teaches recursion — REQUIRED

📘 Static output (educational print trace) — Up to 2 points

✨ Dynamic / interactive / animated output — 2 or more — Impress us!

📚 Full citation of all external tools and sources — REQUIRED

🤖 Final Class Presentations

💡 The top projects (based on creativity, clarity, and educational value) will be presented during the final lecture. This is your chance to showcase your creativity and help shape how future students learn recursion!

🗓 Due Date

Submit your completed project by your section's final week class time:

- TTh Sections: Tuesday, April 27

- MW Sections: Monday, April 28

No late submissions accepted for extra credit.