University of Technology, Jamaica

Faculty of Engineering and Computing

School of Computing & Information Technology

CIT3003: Analysis of Algorithms

Semester 1, Academic Year 2025/2026

Student Group Project and Rubric: Retirement Investment Optimization Using Algorithmic Design

Lecturer: Dr. Arnett Campbell

Tutors: Dr. Arnett Campbell, Mr. Oral Robinson

Date Given: Week of October 20, 2025

Due Date: Week of November 18, 2025

Prepared by Dr. Arnett Campbell, School of Computing & IT

Project Description

This group project challenges students to apply algorithmic design principles to a financial optimization problem. Students will design, implement, and analyze algorithms that model retirement savings growth and expense optimization using **Successive Approximation** and **Binary Search** methods.

The project emphasizes computational efficiency, correctness proofs, and asymptotic analysis, integrating theory with real-world applications. Students may implement their solution in **Python (preferred)** or **any other programming language. They must** use at least one **AI-assisted tool** (e.g., ChatGPT, Gemini, Copilot) to support documentation or design justification.

> **Note on LLM Use:** Students are encouraged to use Large Language Models (LLMs), such as ChatGPT, to assist with brainstorming algorithmic variations, comparing design alternatives, and improving documentation. However, the theoretical justification, proofs, and complexity analysis must reflect original reasoning. Use of an LLM does **not** compromise the evaluation under Theory Criterion #5 (Asymptotic Efficiency Analysis),

provided the student demonstrates their own understanding of computational trade-offs.

Scenario

AofA Financial Services Ltd. seeks an algorithmic tool to forecast retirement investment growth and determine sustainable annual withdrawals. Students must simulate both fixed and variable interest rates, compute post-retirement depletion, and apply Successive Approximation to find the optimal withdrawal amount that reduces the account balance to zero at the end of the lifespan.

Core Tasks

1. **Fixed Growth Simulation** – Implement `fixedInvestor()` for constant-rate growth.

2. **Variable Growth Simulation** – Implement `variableInvestor()` for dynamic rates.

3. **Retirement Expense Simulation** – Implement `finallyRetired()` for withdrawals and returns.

4. **Optimization via Successive Approximation** – Implement `maximumExpensed()` using Binary Search to find the ideal withdrawal.

Function Specifications

1. fixedInvestor(principal, rate, years)

Purpose: Simulate compound growth of retirement savings over a fixed interest rate and time period.
Inputs:

- `principal`: initial investment or contribution amount

- `rate`: annual interest rate (e.g., 0.05 for 5%)

- `years`: number of years until retirement

Output: Total accumulated balance after all contributions and compounding.

2. variableInvestor(principal, rateList)

Purpose: Model the effect of varying annual interest rates (positive or negative) over time.
Inputs:

- `principal`: starting investment value
- `rateList`: list of annual percentage growth rates

Output: Final accumulated balance after applying each year's rate sequentially.

3. finallyRetired(balance, expense, rate)

Purpose: Determine how long retirement funds last under steady annual withdrawals.
Inputs:

- `balance`: starting retirement account value
- `expense`: annual withdrawal amount
- `rate`: expected post-retirement interest rate

Output: Number of years until the balance reaches zero (or negative), indicating depletion.

4. maximumExpensed(balance, rate)

Purpose: Use **Successive Approximation** or **Binary Search** to find the maximum sustainable annual withdrawal such that the balance approaches zero after the modeled retirement period.
Inputs:

- `balance`: initial retirement fund balance
- `rate`: fixed or average growth rate

Output: Estimated optimal withdrawal value.

Illustrative Example

Year	Formula	Example (Salary = $50,000, 5% Savings, 5% Match)	Total
Year 1	salary × (savings + company contribution)	50,000 × 0.15	7,500.00
Year 2	salary × (savings + company) + Y_1 × 1.05	50,000 × 0.15 + 7,500 × 1.05	15,375.00
Year 3	salary × (savings + company) + Y_2 × 1.05	50,000 × 0.15 + 15,375 × 1.05	23,643.75

Assumptions: Employee contribution = 5%; Employer match = 5%; Employer base contribution = 5%; Growth rate = 5% p.a.

Computational Means and Algorithm Category

This project demonstrates the **Divide-and-Conquer** paradigm using Binary Search and **Successive Approximation** for precision. Algorithms operate in **Polynomial Time** (O(n log n)) and emphasize **Efficiency, Correctness, and Scalability**.
Providing this section does **not** compromise the theoretical component; rather, it standardizes the computational model expected while leaving analytical depth (proofs, asymptotic reasoning, and trade-offs) to individual interpretation.

Required Deliverables

- **Theory Component (37 Marks)** – Algorithm Classification, Design, Proof, and Complexity Analysis.

- **Practical Component (75 Marks)** – Implementation, Correctness, Usability, Documentation, and Innovation.

- **Bonus (+5%)** – Real-World Financial Data or Visualization Integration.

Additional Instructions:

Your completed project should summarize the problem in your own words, show the computational means and design technique you decided to use, show the design of the algorithm you used to solve the problem, prove the correctness of the algorithm, and analyze the algorithm to show how efficient and appropriate it is. You should also provide the documentation (theory component), the source code, and the application's run-time executable. (Your project files must be uploaded to the UTech Online/Moodle site for this project.)

The project should be appropriately documented, including the design, implementation, and sample runs. You can run your solution locally or host your solution on Microsoft Azure or a similar cloud platform for demonstration. Still, you are required to upload your submission for grading to the UTech Online (Moodle) course portal. Your tutor will mark your project during the tutorial session when you conduct your project interview. Extra marks will be considered for Real-World Financial Data or Visualization Integration.

Your completed project must run, and you must attend an interview with your tutor during the tutorial class time assigned to receive a project grade. Bring a copy of this project sheet to the interview and place your name and ID number in the provided spaces. **Projects that do not run cannot receive more than 49%, but may receive a much lower grade.**

During the interview, your tutor may ask you to run your application with various user input configurations. Your application must be flexible enough to do so.

No individual projects will be accepted, nor will projects with more than five (5) members be accepted. Plagiarism is a serious offense and will be penalized as outlined in UTech's Student Handbook.

CIT3003 Group Project – Marking Framework

The overall project consists of two major components:

- **Theory Component – 37 Marks (Individual Work)**
- **Practical Component – 75 Marks (Group Work)**

These two components correspond to the **10% (Theory)** and **20% (Practical)** course weightings in the overall CIT3003 assessment.

---

1. Theory Component (Total = 37 Marks)

This section evaluates conceptual understanding, problem formulation, algorithmic design, and analytical reasoning.

Criterion	Description	Marks
1. Problem Identification and Description	Clear statement of the problem, demonstrating understanding of context, assumptions, and constraints.	**5 marks**
2. Algorithmic Classification	Proper identification and justification of the computational class (P, NP, NP-hard, etc.).	**5 marks**
3. Algorithmic Design and Representation	Logical and coherent design with structured pseudocode and detailed explanation of steps.	**10 marks**

4. Proof of Correctness	Validation of correctness using appropriate techniques such as induction, test cases, or reasoning.	**7 marks**
5. Asymptotic Efficiency Analysis	Accurate and well-reasoned time and space complexity analysis with trade-off discussion.	**5 marks**
6. Documentation and Presentation	Clarity of writing, organization, and adherence to formatting and submission standards.	**5 marks**

Total: 37 Marks

---

2. Practical Component (Total = 75 Marks)

This section evaluates the functionality, completeness, and creativity of the implemented solution.

Criterion	Description	Marks
1. User-Friendly Application Interface (CLI/GUI)	Inputs and outputs are clearly presented. The interface is intuitive, visually coherent, and facilitates smooth user interaction.	**10 marks**
2. Correct Operation of the Algorithm in Solving the Problem	The algorithm accurately solves the defined problem, handling all inputs and producing correct results under test scenarios.	**25 marks**

3. **Algorithm Appropriately Represented in Code**	Implementation faithfully follows the algorithmic design; functions, classes, and methods are logically structured.	**10 marks**
4. **Appropriate Error Handling**	The code properly handles invalid inputs, missing data, and other runtime exceptions without crashing.	**10 marks**
5. **Mandatory Use of LLM-Based AI (Copilot, ChatGPT, Gemini, etc.)**	Demonstrated appropriate use of an AI model for design support, debugging, or analysis, with acknowledgment included.	**10 marks**
6. **Innovation, Creativity, and Use of Cloud (e.g., Microsoft Azure)**	Unique and creative implementation with extra features, aesthetic refinement, or deployment to Azure/cloud platform.	**10 marks**

Total: 75 Marks

---

3. Integration into Course Grading - Illustrative Example (Total = 100 Marks)

Component	Course Weight	Mark Base	Conversion Example
Theory (Individual Assignment)	10%	37 marks	30/37 = 81% → Recorded as 81% in individual assignment column

Component	Course Weight	Mark Base	Conversion Example
Practical (Group Project)	20%	75 marks	60/75 = 80% → Recorded as 80% in group project column

---

4. Notes for Assessment and Feedback

- Each section is graded using the mark-based rubric, not weighted percentages.
- Markers must provide concise justifications for awarded marks per criterion.
- Students will receive numerical and percentage feedback for both components.

CIT3003 Theory Component – Easy Grading Rubric (Landscape Format)

This rubric is designed for straightforward use by assessors and provides quick evaluation levels for each major criterion in the theory component..

Criteria	Excellent (4 pts)	Good (3 pts)	Fair (2 pts)	Poor (1 pt)	Weight (Marks)
1. Problem Identification and Description	Problem clearly defined, with a full understanding of the context, assumptions, and constraints.	Problem mostly clear; minor gaps in context or assumptions.	Basic problem description: lacks precision or context.	Problem unclear or incorrect.	5
2. Algorithmic Classification	Correct identification and justification of algorithm class (e.g., P, NP, NP-hard).	Mostly correct classification with partial justification.	Classification attempted but incomplete or weakly justified.	Misclassification or missing justification.	5
3. Algorithmic Design and Representation	Clear, logical pseudocode and design structure; aligns fully with the stated problem.	Good design and structure with minor logical flaws.	Some design clarity, but lacks completeness or precision.	Poorly structured or missing algorithmic design.	10

4. Proof of Correctness	**Strong validation using mathematical reasoning, induction, or test evidence.**	**Reasonable correctness proof; minor errors in logic.**	**Attempted but partially valid or unsupported proof.**	**No correctness proof or incorrect reasoning.**	**7**
5. Asymptotic Efficiency Analysis	**Comprehensive and accurate time and space complexity analysis with clear justification.**	**Generally accurate with minor analytical errors.**	**Partial or vague analysis of complexity.**	**No or incorrect efficiency analysis.**	**5**
6. Documentation and Presentation	**Exceptionally clear, well-organized, professional formatting with strong logical flow.**	**Clear organization; minor formatting or clarity issues.**	**Acceptable presentation but inconsistent structure.**	**Disorganized, incomplete, or difficult to follow.**	**5**

Total Theory Marks: 37 Marks

Notes for Assessors:

- **Use the 4-point scale per criterion and multiply by the Weight ÷ 4 to determine marks earned.**
- **The theory component reflects individual performance and critical understanding.**
- **Encourage analytical justification and professional documentation.**

CIT3003 Practical Component – Easy Grading Rubric (Landscape Format)

This rubric is designed for straightforward use by assessors and provides quick evaluation levels for each major criterion in the practical component.

Criteria	Excellent (4 pts)	Good (3 pts)	Fair (2 pts)	Poor (1 pt)	Weight (Marks)
User-Friendly Application Interface (CLI/GUI)	Highly intuitive interface; clear I/O layout, smooth navigation, professional aesthetics.	Mostly clear interface with minor usability or design issues.	Functional but cluttered or confusing interface; limited polish.	Interface is incomplete, disorganized, or difficult to use.	**10**
Correct Operation of the Algorithm in Solving the Problem	Produces fully correct results under all test conditions; fully meets objectives.	Minor inaccuracies under limited conditions; generally meets objectives.	Partial success in solving the problem; inconsistent or incomplete results.	Fails to solve the problem or produces incorrect outputs.	**25**
Algorithm Appropriately Represented in Code	Code design fully reflects the algorithmic logic and theoretical structure; clean, modularized.	The code mostly demonstrates the design, though it has minor structural inconsistencies.	Partial reflection of design; code structure needs improvement.	Code is poorly structured or diverges from the algorithmic plan.	**10**

Appropriate Error Handling	Comprehensive error management; handles all invalid inputs and exceptions gracefully.	Handles most errors with minimal issues, or misses some boundary cases.	Limited error management; inconsistent handling of exceptions.	No error handling; frequent crashes or undefined behavior.	**10**
Use of LLM-Based AI (Copilot, ChatGPT, Gemini, etc.)	Clear, ethical use of AI tools for validation, debugging, and documentation; fully acknowledged.	Effective AI use, but missing acknowledgment or a minor ethical lapse.	Minimal or inconsistent AI use; lacks clear documentation.	No or unethical use of AI tools.	**10**
Innovation, Creativity, and Use of Cloud (e.g., Microsoft Azure)	Demonstrates strong creativity, unique problem-solving, or successful cloud deployment.	Some creative elements or partial cloud implementation.	Limited creativity or cloud usage; minimal extension beyond requirements.	No innovation or cloud component included.	**10**

Total Practical Marks: 75 Marks (including all weighted criteria)

Notes for Assessors:

- Use the 4-point scale for each criterion, and multiply each score by its **Weight** divided by 4 for proportional marking.

- **Projects that fail to execute score ≤49%.**
- Bonus marks may apply for Real-World Financial Data or Visualization Integration.