The Actual 9569 Outcomes

These are the computing syllabus outcomes, you can use this to check your readiness.

Section 1: Algorithms and Data Structures

1.1 Algorithmic Representation

- Employ pseudo-code and flowcharts to demonstrate the flow of programs.
- Utilize standard symbols in flowcharts.
- Incorporate various control structures in algorithm design.
- Use decision tables to examine possible actions based on different input combinations (up to three conditions).
- Apply modular design principles to break down problems into manageable parts.

1.2 Fundamental Algorithms

- Code sorting algorithms like insertion sort, bubble sort, quicksort, and merge sort.
- Explain sorting algorithms through practical examples.
- Implement search algorithms including linear search, binary search, and hash table search.
- Illustrate how search algorithms work using examples.
- Analyze and describe the efficiency of sorting and searching algorithms with Big-O notation for time complexity (excluding space complexity).

1.3 Data Structures

- Comprehend and implement algorithms for stacks and queues (both linear and circular), as well as linear linked lists and binary search trees.

- Understand static and dynamic memory allocation.
- Perform operations like creation, insertion, and deletion for stacks and queues.
- Recognize the role of a free space list in memory management.
- Conduct create, update, and search operations on linear linked lists (excluding doubly-linked and circular versions).
- Perform create, update (excluding node deletion), and search operations on binary search trees.
- Understand tree traversals (pre-order, in-order, post-order) and specifically apply in-order traversal for binary search trees.

Section 2: Programming

2.1 Coding Standards

- Apply common coding standards, focusing on indentation, naming conventions, and commenting (including programmer name, date, program description, and version control).

2.2 Programming Elements and Constructs

- Understand and initialize variables of types like integer, real, char, string, and Boolean, and manage arrays.
- Utilize common library functions for input/output, string manipulation, and mathematical calculations.
- Apply fundamental programming constructs for sequence, selection, and iteration.
- Use functions and procedures to segment code into functional modules.
- Comprehend and implement recursive algorithms.
- Trace and record outcomes of recursive and non-recursive programs.
- Explore the use of stacks in recursive programming.

2.3 Implementing Algorithms and Data Structures

- Code sorting algorithms (insertion sort, bubble sort, quicksort, merge sort) and search algorithms (linear search, binary search, hash table search).
- Develop programs for data structures like stacks, queues, and linked lists (excluding doubly-linked and circular versions).
- Manage data storage and retrieval in text files.

2.4 Data Validation and Program Testing

- Differentiate between data validation and verification.
- Learn and apply various data validation techniques.
- Identify and resolve different types of programming errors (syntax, logic, runtime).
- Design and implement test cases using normal, abnormal, and extreme data sets for software testing and debugging.

2.5 Fundamentals of Object-Oriented Programming

- Define and understand the concepts of classes and objects.
- Explain encapsulation and its benefits in information hiding and implementation independence.
- Understand inheritance and its role in promoting software reuse.
- Explore polymorphism and its application in code generalization (excluding method overloading and multiple inheritance).

Section 3: Data and Information

3.1 Data Representation

- Represent data in binary and hexadecimal forms.
- Write programs to convert positive integers between denary, binary, and hexadecimal bases.

3.2 Character Encoding

- Provide examples of Unicode usage.

- Incorporate ASCII code in programming.

3.3 Databases and Data Management

- Define database elements like tables, records, and fields.
- Use and explain different types of database keys.
- Discuss concepts of data redundancy and dependency, reducing redundancy to third normal form.
- Create ER diagrams to illustrate table relationships.
- Compare SQL and NoSQL databases in addressing database management challenges.
- Program with both SQL and NoSQL databases.
- Understand data privacy and integrity protections.
- Explain data backup versus archiving, and the importance of version control and naming conventions.
- Discuss data protection under the Personal Data Protection Act in Singapore.

3.4 Social, Ethical, Legal and Economic Issues

- Understand the ethical conduct required of computing professionals.
- Discuss the impact of computing on social and economic aspects of lifestyle and workplace.
- Analyze the broader social, ethical, legal, and economic implications of computing technologies.

Section 4: Computer Networks

4.1 Fundamentals of Computer Networks

- Explain network concepts such as LAN, WAN, intranet, and internet structure.
- Understand IP addressing and DNS.
- Discuss the necessity of communication protocols in networks.
- Describe data transmission in packet-switched networks.

- Explain client-server architecture and implement simple server-client interactions using socket programming for specific scenarios like a tic-tac-toe game.

4.2 Web Applications

- Differentiate between web applications and native applications.
- Apply usability principles in designing web applications.
- Develop web applications using HTML, CSS, and Python to handle tasks such as:
  - Accepting user inputs like text and image files.
  - Processing these inputs on a local server.
  - Storing and retrieving data.
  - Displaying outputs in formats such as formatted text, images, or tables.
- Conduct testing of web applications on a local server.

4.3 Network Security

- Understand the threats posed by malware (like worms and viruses) and denial of service (DoS) attacks, and their impact on computer systems.
- Recognize the roles and limitations of security measures such as firewalls, intrusion detection systems (IDS), and intrusion prevention systems (IPS) in restricting network access.
- Comprehend how encryption, digital signatures, and authentication contribute to the security of network applications.