**Revision Guide: Algorithm Design for Any Given Scenario**

### **Overview**

This guide is designed to help you understand the key aspects of algorithm design, applicable to any scenario you may encounter. It covers decomposition, algorithm features, and assessment criteria to support your learning and preparation.

### **1. Decomposition Coverage**

When designing algorithms, focus on selecting key processes that meet the needs of the specific scenario. Some key problems to consider may include:

* **Handling User Access:** This includes processes such as setting up accounts, logging in, and changing passwords.
* **Creation and Processing of Data:** Consider how to manage data relevant to the scenario, such as tracking information, managing scores, or gathering preferences.
* **Communication/Data Exchange:** Focus on how to analyse data to make recommendations, or how to integrate information from different systems.
* **Key Calculations:** Include calculations related to time, costs, or any metrics relevant to the scenario.
* **Data Filtering and Visualisation:** This could involve various visual aids such as decomposition diagrams or navigation maps.

Decomposition can be represented through descriptions or visual aids such as decomposition diagrams or navigation maps. Aim for highly effective decomposition that comprehensively covers the required inputs, processes, and outputs.

### **2. Algorithms**

Your algorithms may take various forms, including:

* **Flowcharts:** Visual representations of the algorithm's steps.
* **Pseudocode:** A structured way to outline the logic of the algorithm in plain language.
* **Data Flow Diagrams (DFDs):** Diagrams that illustrate how data moves through the system.

Expected features of your algorithm design should include:

* Clearly defined steps that depend on the input and the results of preceding steps.
* A logical flow that is easy to follow.
* Clearly identified output types, e.g., output to screen, physical printouts, or returns to functions.
* A modular approach, incorporating subroutines, procedures, and loops as appropriate.
* An efficient design that avoids unnecessary steps.
* The use of comments and resources like APIs, CSV files, or databases clearly identified.
* Attention to detail and logical dependencies.

Your algorithm designs should not aim to showcase the entire solution but rather illustrate how you would solve several key problems, limiting your designs to a maximum of five complex issues.

### **3. Assessment Criteria**

Your work will be assessed based on the following focus areas:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Assessment Focus** | **Band 0** | **Band 1** | **Band 2** | **Band 3** |
| **Decomposition of Problem** | No rewritable material | Basic decomposition covering required processes and outputs | Good decomposition covering required processes and outputs | Highly effective decomposition covering all required processes and outputs |
| **Application of Logical Thinking and Conventions** | Some effective use of accepted conventions, but with inconsistencies | Mostly correct outcomes due to mostly logical logic and correct structure | Consistently correct outcomes due to precise logic and correct structure | Highly effective and consistent use of accepted conventions |

### **Tips for Success**

* Review examples of decomposition and algorithm design to enhance your understanding.
* Practice creating flowcharts and pseudocode for various hypothetical scenarios to solidify your skills.
* Familiarise yourself with data flow diagrams and how they help visualise data movement.
* Ensure your algorithms are modular and efficient, minimising unnecessary steps.

This guide should serve as a solid foundation for your understanding of algorithm design in any given scenario. Best of luck with your revision!