Competitive Programming Primer: Design Document

Members:

Pedro Marcet (pmarcet2023@my.fit.edu)
Jon Ayuco (jayuco2021@my.fit.edu)
Ivan Marriott (imarriott2023@my.fit.edu)

Advisor: Dr. Raghuveer Mohan (rmohan@fit.edu)

Table of Contents

1.0 INTRODUCTION

1.1 Purpose

The *Competitive Programming Primer: Design Document* describes the architecture and system design of the ICPC Database, the Manim Data Visualizer, and the Hackpack at an early stage of their development. This document is meant to be viewed and edited by the Competitive Programming Primer development team, and viewed by any other interested parties.

1.2 Scope

Provide a description and scope of the software and explain the goals, objectives and benefits of your project. This will provide the basis for the brief description of your product.

The Competitive Programming Primer is made up of three components: the Manim Code Visualizer, the ICPC Database, and the Hackpack. Each component is separate, but together add to the goal of setting up tools that will aid any student interested in competitive programming. As each component is separate, the scope of each component will be described as so.

The Manim Code Visualizer is to be a visual aid for competitive programming members and computer science students to effectively learn different forms of data structure.

The ICPC Database is a website that aggregates every problem in ICPC's tournament

1.3 Overview

Provide an overview of this document and its organization.

1.4 Reference Material

This section is optional.

List any documents, if any, which were used as sources of information for the test plan.

1.5 Definitions and Acronyms

This section is optional.

Provide definitions of all terms, acronyms, and abbreviations that might exist to properly interpret the SWDD. These definitions should be items used in the SWDD that are most likely not known to the audience.

2.0 SYSTEM OVERVIEW

Give a general description of the functionality, context and design of your project. Provide any background information if necessary.

3.0 SYSTEM ARCHITECTURE

3.1 Architectural Design

Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high level overview of how responsibilities of the system were partitioned and then assigned to subsystems. Identify each high level subsystem and the roles or responsibilities assigned to it. Describe how these subsystems collaborate with each other in order to achieve the desired functionality. Don't go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together. Provide a diagram showing the major subsystems and data repositories and their interconnections. Describe the diagram if required.

3.2 Decomposition Description

Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an objectoriented (OO) description. For a functional description, put toplevel data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

3.3 Design Rationale

Discuss the rationale for selecting the architecture described in 3.1 including critical issues and trade/offs that were considered. You may discuss other architectures that were considered, provided that you explain why you didn't choose them.

4.0 DATA DESIGN

4.1 Data Description

Explain how the information domain of your system is transformed into data structures. Describe how the major data or system entities are stored, processed and organized. List any databases or data storage items.

4.2 Data Dictionary

Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

5.0 COMPONENT DESIGN

In this section, we take a closer look at what each component does in a more systematic way. If you gave a functional description in section 3.2, provide a summary of your algorithm for each function listed in 3.2 in procedural description language (PDL) or pseudocode. If you gave an OO description, summarize each object member function for all the objects listed in 3.2 in PDL or pseudocode. Describe any local data when necessary.

6.0 HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

Describe the functionality of the system from the user's perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

6.2 Screen Images

Display screenshots showing the interface from the user's perspective. These can be hand drawn or you can use an automated drawing tool. Just make them as accurate as possible. (Graph paper works well.)

6.3 Screen Objects and Actions

A discussion of screen objects and actions associated with those objects.

7.0 REQUIREMENTS MATRIX

Provide a crossreference that traces components and data structures to the requirements in your software requirements specification (SWRS) document.

Use a tabular format to show which system components satisfy each of the functional requirements from the SWRS. Refer to the functional requirements by the numbers/codes that you gave them in the SWRS.

8.0 APPENDICES

This section is optional.

Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.