### FINAL PROGRAMMING PROJECT

### CS 172 Computer Science II

|  |  |  |
| --- | --- | --- |
| **CATEGORY** | **POINTS** |  |
| Proposal |  | 5 |
| Requirements Specification &  Project Management Plan |  | 20 |
| System Implementation |  | 50 |
| PowerPoint & Presentation |  | 15 |
| Individual Team Member Summary |  | 10 |
| **TOTAL** |  | 100 |

* This is intended to be a **team project (2 – 3 team members)**.
* There is **NO Late Acceptance** of any of the requirements

***DELIVERABLES***

* **Project Proposal (5 pts)**

**Due Thursday, November 9, 2017 – 1 document per team**

* + List all **team partner**(s) and a **Name** for your team.
  + Provide a **clear problem definition**. What is required of the system? What must it accomplish or provide? Do you make any assumptions?
* **Requirements Specification and Project Management** **(20 pts) BEFORE CODING!**

**Due Thursday, November 16, 2017 – 1 document per team**

* + Essentially the design for your project. **Complete UML diagrams** of classes, member variables, member functions, and relationships between classes. What classes, structures, vectors, variables, functions, etc. will be required?
  + How will you begin to approach the problem? What are the anticipated challenges?

**Project Management Plan**

* Name **individual team member tasks/responsibilities**. Split up the implementation into portions and tasks per team members.
* **Timeline** of individual deliverable dates for deliverables and team events
* **System Implementation & Presentation/Demo & Individual Summary**

**Due Tuesday, December 12 during Class Finals Timeslot**

1. **Project Implementation (50 pts)**

* **Your project implementation** **MUST** include the following:
  + - **Object-oriented programming with** **classes**
    - **Separate headers, implementation files** and a **main driver**
    - **File I/O. Include some form of file use even if only to capture results.**
    - **Well-commented code, “pretty code” (indentation and spacing to make it easily legible)**.
    - **Sections commented with the developer’s name**
    - **Input validation and Well-tested code.**
* You should **employ** the following concepts **as appropriate** to the program: vectors, pointers, dynamic memory. Other possibilities include inheritance, polymorphism, recursion, advanced data structures of the STL Library(e.g linked lists, stacks, queues, binary trees, etc.)
* **To make an A on this project**, you must reach above and beyond in concept and code sophistication.

1. **Project Presentation & Demonstration (15 pts)**
   * **5 – 10 Minutes max for presentation and demo!**
   * Present in a **professional, polished style** as though in industry.
   * **PowerPoint** or presentation package to include (**Maximum 6 slides**):
     + Problem definition
     + Itemized requirements and challenges
     + Class hierarchy diagram (or UML Class Diagram)
     + Brief overview of general solution approach
     + Distinctive aspects of your approach (the “cool” factor)
   * Brief **demonstration** of working program
2. **Individual Team Member Summary (10 pts)**
   * Due in **hard-copy** form, an individual participation write-up from each student.
   * What did you contribute to the project?
   * How did the team work together?
   * What did you learn from the project?
   * What you would do differently on another project?

**//////////////////////////////////////////////////////////////////////////////////////////////////////**

### What to do? … Some Project Ideas

***Feel free to use those bright, creative minds – think about it, get a proposal together!*** The following are some ideas for final projects. **You are not limited to these ideas.** Be creative and bring ideas to your instructor! Your problem should be of equal or greater complexity than these suggested problems. You can also look through advanced problems in your text or other texts for more ideas. You can take a problem and add features, enhancements.

* **Model/simulation** – Simulations make for a great project because they present real, practical problems easily modeled yet with complexity. Implement a program that simulates a real-life scenario. Examples are simulating an airport air traffic control scheduling, processes of a hospital, a manufacturing floor, movement of aid for a natural catastrophe,
* **Education: Interactive tutoring project** – Implement a program that creates an interactive tutoring for children or high school students. Include some degree of “intelligent” tutoring – in other words, adjust your questions or difficulty of questions according to user responses and the system should have a level of sophistication. **Games with educational purposes.** Types of games are increasingly used as educational tools. For instance a race to the finish line (requires providing correct answers, etc.). A version of Monopoly with both “rolled dice”(random generation) and correct answers, is another that is used increasingly.
* A **Business application** - For instance, a system for a small business might use a number of files and calculate profits (as per purchases, sales, expenses), might calculate payroll, keep track of bank accounts, etc.

**Pizza Business.** A system to process customer orders, how many pizzas per order, how many orders? What has been ordered, delivered, still in the works? What toppings, sizes, crusts? What costs per pizza? You could also have other aspects of the business, payroll, employee shifts, supply costs, profits, etc.

You could have a pizza class, an order class and a structure holding orders (i.e. vector, dynamic array, etc.)

**Gas Station Business.** Model a gas pump. Customer interaction – pump gas and amount to be paid. Track sales of gas, purchases of gas – how are your pumps? Getting low of gas, need to order more? Sales and profits. Maybe payroll.

* **Explore** other advanced problems in your text or other texts or in texts that I can make available.

Consider aproject involving more advanced data structures (e.g. linked lists, stacks, queues, binary trees, etc.)

* **A text-based game** of your choice.
* **An Interdisciplinary Application**. For instance, a physics model, a bioinformatics model, a biological model, a societal or anthropological issue, scientific problems, an economic model.

***Grading Criteria for Code Implementation:***

|  |  |
| --- | --- |
| 20% | **Problem analysis**: Does the design for the implementation solution exhibit a general understanding of the problem definition? |
| 30% | **Execution and Results**: The code meets the stated minimum implementation requirements. Does the project **work** correctly and generate the correct answer(s) according to the problem specifications? |
| 10% | **Skill Requirements:** Does the code meet the stated minimum implementation requirements? |
| 20% | **SW Development Quality**: Does the project utilize appropriate and effective programming concepts? Is the project well designed, contains comments and contains clean, well-organized code. Is the code readable and easy to follow? Commented name of developer for blocks of code |
| 20% | **Degree of problem difficulty:** How challenging is the problem?  **Above and Beyond:** Top projects designs show ingenuity, creativity, and efficiency. Demonstration that you have shown initiative, going above and beyond basic course coverage into utilization of advanced concepts. |