Project1 Design
Dargo Wang

1. Analysis:

Use case	Starter
Description	When a user enters the system, it will be promoted to enter its name.
Actors	Any user
Assumptions	After the user enter its name, the system will show “Welcome (name)”
Steps	1. Prompt the user to enter its name 2. Display “Welcome (name)”
Issues	

Use case	Main menu
Description	After welcoming the user, the system gives out a list of options for user to choose what it wants to do. If the user enters 1, it will enter the game mode. If the user enters 2, it will enter the viewer mode. It the user enters 3, it will quit.
Actors	Any user
Assumptions	After the user choose one from the options, the system will turn into that part.
Steps	1. Give a list of options for user to choose from 2. Prompt the user to choose 3. Turn into the option
Issues	

Use case	Viewer mode
Description	If the user choose option 2 in main menu, the system turns into viewer mode and display list of high scores.
Actors	Viewer
Steps	1. Display the list of top high scores 2. Back to main menu
Issues	What if there is less than 10 players so far

Use case	Game mode
Description	After the user chose to enter the game, the system will display the character and then enter the first cycle.

	In each cycle, the system displays how many steps the player is from the goal and time left; then provide a list of options for player to choose.
Actors	Player
Assumptions	After the player choose one from the options, the game will execute that functions.
Steps	1. Display current character. 2. REPEAT: 2.1 Display steps from the goal, and time left. 2.2 Give a list of options for user to choose from 1) Move forward 2) Read technical papers 3) Search for loose change 4) View character 5) Quit the game 2.3 Prompt the user to choose 2.4 Turn into the option 3. When the time has fallen to 0, the player dies. 4. Print score and adjust high scores.
Issues	A special function – time left

Use case	Puzzle
Description	This is a possible part when player chooses option 1 (move forward)
Actors	Player
Assumptions	After the user choose one from the options, the system will turn into that part.
Steps	1. Display the message of PUZZLE 2. Provide a list of options for player to choose from 3. Prompt the player to choose 4. Adjust intelligence according to the answer
Issues	

2. Design:

Class name	**Project1**
Purpose	Serves as the system that execute the main function, driving the whole game.
Member variables	(String) name;

Functions	main();
	mainMenu();
	gameMenu();

	moveForward();
	readPaper();
	searchChange();
	viewCharacter();
	quit();
	die();

Classes depend on	/
Classes use	HighScores; Encounters

Class name	**HighScores**
Purpose	Keep and adjust top 10 high scores
Member variables	(string[]) ArrayOfName; (int[]) ArrayOfScore; (int) newScore; (int) curNum; (int) MAX_NUM;

Functions	display();
	insertScore();
	insertName();
	readFile();
	writeFile();

Classes depend on	Project1
Classes use	Project1 Encounters

Class name	**Characters**
Purpose	Store and adjust all characters of the game
Member variables	(int) time; (int) intelligence; (int) money; (int) stepsFromGoal; (int) score;

Functions	changeTime();
	changeIntelligence();
	changeMoney();
	changeSteps();
	changeScore();
	getTime();
	getIntellegene();
	getMoney();

	getSteps(); getScore();
Classes depend on	Project1; Encounters;
Classes use	HighScores

Class name	**Encounters**
Purpose	Provide all encounters the player will have while choosing from an option in the game mode.
Member variables	

Functions	chooseEncounter();
	nothing();
	puzzle();
	professor();
	student();
	gruntWork();
	paperToGrade();

Classes depend on	Project1
Classes use	/

3. <u>Test:</u>

1. **Name enter test**: 1 test. Test if a user enters his name, the system will print correct welcome message.

2. **Main menu tests**: 3 tests. When a user chooses 1 of 3 options in main menu, the system will turn into correct mode.

3. **Adjust score tests**: 4 tests. When a user dies his game, the score will be inserted in correct location in the list, if his score is top 10; otherwise, no change to the list.

4. **Encounter tests**: n tests. Test if each encounter is doing the right thing as designed.

5. **Puzzle tests**: n tests. Test the reaction after entering the correct/wrong answer in the puzzle.