1. **2. Requirements [20 marks]:**
    a. a) Write a succinct introduction explaining how requirements were elicited and negotiated, and why they are presented as they are. Your submission should **evidence research into requirements** specification and presentation (4 marks, ≤ 1 page).
    b. b) Give a systematic and **appropriately-formatted** statement of requirements, including, for each requirement, a note of any relevant environmental assumptions, associated risks, or alternatives (16 marks, ≤ 3 pages).

Plan

User Requirements

- Cross platform
- Boats with different attributes, need to be balanced
- Cpu respect rules of game
- Difficulty cannot be selected, gets harder with each leg
- Stamina resets
- Ai gets smarter with each leg, smarter in terms of avoiding obstacles (equations?)
- Static and dynamic obstacles
- Penalty is fixed, time added to leg time
- Splash dialogue that explains the game before the first leg
- Minimum of 3 teams
- 2D

Additional User Requirements (basically ignore)

- Power ups to repair, speed up boat
- Music (not copyrighted)
- Score board?

1. **Single Statement of Need (SSON)**
    a. "The game should be able to be played and enjoyed by the user"
    # if you don't like it you can change it

List of functional requirements:

Transformation

- When user pressed keys, the boat avoids obstacles
- Timer displays how long the leg is taking
- Screen will display obstacles when they are meant to appear
- The robustness will decrease when an obstacle is hit

Invariant

- Have a boat that the player can control

- Have other boats that can be controlled by cpu
- Have obstacles that need to be avoided
- Have attributes for each boat (speed, acceleration, maneuverability, and robustness)
- Increasing difficulty

Failures

- When the user presses a different key, it wont crush. Prevents wrong user input.
- Screen should display game over when robustness reaches 0
- Time limit? Otherwise could go on forever
- Should not be allowed to stay in another players lane for the entire game
- Should not be allowed to move backwards
- Should not have a way of “cheating”

List of non-functional requirements

- • **Security**
  - not necessary since the user can access whenever he/she feels like (do we even need authentication?) not really, maybe just like make sure they cannot change the code? How i mean in theory you should be able to access it anyway i mean that’s how steam workshop works or modding. Basically we can say java uses a compiler so we just give the executable file rather than the source code, therefore secure tada # ok kinda
- • **Reliability/Availability**
  - the game is always available if the minimum system requirement are fulfilled and the electricity bill is paid
- • **Timing**
  - There will be a loading screen(Temporal constraint) where the control would be shown to inform the user how to play the game.
- • **Precision constraint**
  - #Error margin percentage
- • **Maintainability**
  - Any JAVA software engineer should be able to work on the source code
- • **Documentation**
  - The game would have all the necessary controls information in the option menu(# where the user can change it?)
- • **Resilience**
  - If the game crashes shouldn't affect the device that is being used
  - If the game fail to load it will force quit the game
- • **Integrability**
  -
- • **Scalability**
  - Used only by university of york and us
- • **Operability**
  - The game should be able to played by anyone including inexperienced player or normie, all noobs welcome

    - It should also be intuitive to play
- • **Auditability**
    - The game should keep a record of the player scoreboard(records) and achievements
- • **Accessibility**
    - the system shall be operable by people affected by color blindness.
- • **Usability**
    - The game is pegi 3+ , written in plain english(only)

System Requirements

- Cross platform because programmed in java

Constraint requirements:

• Project constraints:

There is no financial constraint

• Process constraints:

The code is written in JAVA

• Design constraints :

The Game should be able to work on most system # if not please state specification

Write up

The requirements for The York Dragon Boat Race have first been developed through reading the product brief and carefully evaluating the specific features that are mentioned in them. Some of the requirements were explicitly stated like the attributes of the boats and increasing the difficulty, however there were also requirements that were too vague to leave on their own, like the obstacles on the river. After coming up with a list of requirements, together as a team we arranged a Team-Customer meeting to further discuss the requirements with our customer. The meeting with the customer occurred on the [insert date] where we asked a series of prepared questions on the requirements to get more information. Some of the questions we included were:

1. “What is a “penalty”? Should a team be disqualified for breaking the rules? Time restrictions (for how long allowed outside of a lane)?”
2. “Do you have a preference for what the obstacles in the river should be?”
3. “How many teams should compete in each leg?”

After the meeting we came up with the user requirements, which we then divided into mandatory and additional to ensure that the team would stay on track. This was decided with the customer during the Customer Meeting by analysing the customers responses. After finalizing the user requirements, as a team we discussed the system requirements and divided it into function, non-functional and constraint requirements.

1. **User requirements**

ID	Description	Priority
UR_CROSS_PLATFORM	The game should be accessible	Shall

	across all platforms i.e mobile, desktop	
UR_BALANCED_BOATS	The boats in the game should be balanced	Shall
UR_INCREASED_DIFFICULTY	The difficulty will increase each completed leg	Shall
UR_STAMINA_RESET	The boats stamina will reset each leg	Shall
UR_OBSTACLES	There should be static and dynamic obstacles that boats will avoid or else the boats robustness will decrease	Shall
UR_FIXED_PENALTY	The penalty for leaving the designated lane will be fixed each time	Shall
UR_SPLASH_DIALOGUE	There will be splash dialogue explaining the controls	Should
UR_SMARTER_AI	The boats CPU will improve each leg to increase difficulty	Should
UR_CONTROL_BOAT	The user can control the boat using the arrow keys	Shall
UR_WIN_LEG	The winner of each leg will be determined by which boat crosses the finish line with the shortest time	Shall
UR_GAME_OVER	When the robustness of the boat reaches 0 or all the legs are complete, the game is over	Shall

2. Functional requirements

ID	Descriptions	User requirements
FR_CONTROLS	The game will allow the user to move the boat	UR_CONTROL_BOAT
FR_TIMER_LEG	The game will show a timer displaying how long the leg is taking	UR_WIN_LEG
FR_OBSTACLE	The screen will display obstacles when they are meant to appear	UR_OBSTACLES

FR_HP	The robustness of the boat will decrease when an obstacle is hit	UR_OBSTACLES
FR_BOAT_PLAYER	The game will provide a boat that the player can control	UR_CONTROL_BOAT
FR_BOAT_CPU	The game will provide other boats that are controlled by cpu	UR_IMPROVE_AI
FR_NO_BACKWARDS	Boats are not be allowed to move backwards	UR_CONTROL_BOAT
FR_ATTRIBUTES	the boats have attributes for each boat (speed, acceleration, maneuverability, and robustness)	UR_BALANCED_BOATS
FR_DIFFICULTY	The game will progressively increase difficulty	UR_INCREASE_DIFFICULTY
FR_CONTROL_INPUT	The game should not allow wrong user input.	UR_CONTROL_BOAT
FR_GAME_OVER	The screen should display game over when robustness reaches 0	UR_GAME_OVER
FR_OWN_LANE	Boats are not to be allowed to stay in another participant’s lane for the entire game	UR_FIXED_PENALTY

FR_TIMER	During the race there will be a time limit that will be displayed on top of the screen	UR_WIN_LEG
FR_ANTI_CHEAT	The game should not have a way of cheating	UR_CONTROL_BOAT
FR_OFFLINE	The game should be allowed to be played offline if downloaded	UR_CROSS_PLATFORM

3. Non-functional requirements

ID	Descriptions	User requirements	Fit Criteria
NFR_INIT	The game should be able to start instantly	UR_CROSS_PLATFORM	within 3 second after accessing
NFR_LOADING	Fast loading screen	UR_CROSS_PLATFORM	The loading screen will be around 8 seconds
NFR_EZ	The game does not require training to be played	UR_SPLASH_DIALOGUE	the age restriction for the game to be played PEGI +3
NFR_AVAILABILITY	the game will be available all the time to be played	UR_CROSS_PLATFORM	There is no limitation on when to access if not on a hardware level so 99% cases
NFR_CRASH	The game will not affect the device negatively	UR_CROSS_PLATFORM	CPU usage percentage: 30%
NFR_ACHIEVMENTS	The game should allow the player to see his game achievements	UR_SCOREBOARD	Shown at the record option menu
NFR_SCOREBOARD	The game should allow the player to check his statistics	UR_SCOREBOARD	Shown at the record option menu
NFR_FORCE_QUIT	If the game freezes it will close it self	UR_GAMEOVER	30 seconds of frozen screen will result a force quit