# Computer Science Project Proposal

Describe the stakeholders for your project:

For young children who are still developing their minds but could also be for older people. Since children enjoy dynamic things rather than static things like lessons. Games are dynamic and require more input from the user, so children who naturally like to play, will be attracted to games. A lot of children are distracted by games for long periods of time and for my project I think this could be beneficial since the more they play, the more practice they get with problem solving.

List at least 15 success criteria for your project: *(continue overleaf)*

*Static obstacles*

*Trap/fake static obstacles*

*Multiple levels increasing in map difficulty*

*Collectables/obtainable items*

*Controllable player*

*Loading screen with skip button and tips*

Describe your project proposal:

It would be a game where the player is a stick figure who has to pass through obstacles that are of two kinds, the static kind of obstacle that doesn’t move and only affects the player if the player comes into contact with it, or a hidden kind of obstacle that appears to be static, but once the player enters into an area close enough to the obstacle, the obstacle itself tries to eliminate the player and make them lose a life, or health. These two types are intended because it is to test and help develop the player’s logical reasoning, there will be hints to whether an obstacle is a fake or a real, like maybe a button/plate on the ground that can be stepped on by the player, suggesting a trap. There will also be a boss fight that will also test the player’s problem solving skill, an boss who tracks and follows the player continuously, and can only be hit in the back. This is so that the player has to come up with ideas to force the boss to stop facing them, giving the player a chance to win, for example blinding the boss could be possible by having some obtainable items obtained from beating levels that are seemingly abstract, and appear to be useless items, but the player will have to use their reasoning the find creative ways of using the items.

Name: Daichen CUI

Proposed Language: *Python Pygame*

What resources will you use to design and write your project?

Visual Studio Code – to create the game and write all the code there

Pygame.org – resources and libraries to teach how to draw certain obstacles.

Programarcadegames.com – to learn how to add different features, such as collideable platforms

Success criteria continued:

Menu with maybe a shop for cosmetics as to reward the player for playing the game?. And play button, can also be used for pausing game because single player

Background design (with a sun rising and falling to show timer left for certain difficulties)

Disappearing platforms/moving

Platforms that detect player presence (button/plate), as well as collideables

Health + Lives, link with consumables so health and lives can be regained

Win screen to indicate to the player that they beat the level/ loss screen if they ran out of lives

Respawn system for every life used, if their health falls below 0, respawn them from last checkpoint (level)

Make a functioning boss that faces the direction of the player depending on what side of the screen they are on, that can attack, defend etc

Tutorial teaching player controls, etc

Settings, and timers

Describe the influences and inspiration for your project. Talk about the research you have done and how this has influenced the features for your own project.

I went on the computer science trip and experienced some retro games. These games like the ones from the portal series, some of the lego games, they all had logical reasoning behind them and a lot of problem solving. I thought since gaming should be fun, it shouldn’t be useless to the player apart from pure entertainment. Including logical reasoning in a game allows players to both have fun while also benefitting from video games.

**Implementation**

Your project is 20% of your final grade.

Use this page and the next to think about how you would implement some of the algorithms you have mentioned in your success criteria above. Number each one and write the pseudo-code, with comments, for each below.

* Static obstacles could be made using classes where any other object that comes into contact with a class of the obstacle will be unable to pass through, so when the x y coordinates of the player come within the boundaries of the x and y coordinates of the object.
* Moving / fake static obstacles could be part of the class of static obstacles, but should be part of a subclass, where specifically for the fake obstacles, there is a method that is run everytime the player’s character is detected within a certain radius. This method could be a very simple block of code that simply moves the obstacle along an axis, like for example a block that tries to destroy the player’s character, simply having the x-position of the obstacle being incremented will be enough for the object’s movement. Then I will have to have the code constantly check, maybe using a loop whenever the object is in contact with the player. This allows the game to determine whether to damage the player or not, since if the moving obstacle hits the player, they should take damage.
* For both kinds of obstacles, there should be an immunity period for each obstacle where if for example the player is damaged by the obstacle, they cannot be damaged by it again for as long as they are still in contact with the obstacle, because it wouldn’t make much sense to continuously take damage when the obstacle is a static kind as in like a spike for example, this immunity property will not be in the classes for more fluid like objects like lava.
* Dissappearing objects I will have to learn how to make it so that it is only collideable with by the player during the period it appears. Maybe they could be another separate sub class
* Health and lives could be variables apart of the player’s character class. Everytime they take damage, the health is decremented, if it is below or equal to 0, the player’s position gets reset back to the start position and a life is decremented.
* If the lives become 0, a variable which keeps track of the player’s level apart of the player’s class also gets reset therefore bringing the player back to the first level. This is because it will teach the player to learn from their mistakes under pressure so that they don’t do it again.
* The movement of the player can be implemented with an if statement and keydown events on pygame.
* The tutorial will be simple and consist of either the player going through a level 0, where the game’s mechanics are taught by having them being drawn on signs as the player progresses so they can use what they learn immediately, like how to jump and that.
* I will need to find out how to make it so that an obstacle (for the boss) will have one side that continuously faces

**Implementation (continued)**

**Screen Design**

What form will the user interface take? What will be visible to the user (e.g. high score, map, HUD, instructions, etc). Draw a design of the screens you will have in your game below and on the next page. Annotate each screen with an explanation.

I thought that the classic, platformer kind of game, Super Mario series would be my main inspiration of the game



This is a good example because on the top left, you can see how many lives you have indicated by the player’s character’s icon, with the number of lives.

On the top right there is also a score and timer. Mine probably wont have a timer on earlier levels since the objective is to train children’s problem solving ability. Putting them under timed pressure when they haven’t developed the skills yet, is just a waste. So in earlier levels a there will be no timer, to purely develop the skills first, then once they have developed the skills, timers might be applied in certain areas of the game to enhance the skills.

On the top left underneath the lives, instead of having a currency system, I will add an icon that looks like a leather bag, indicating the player’s inventory where they can see their possessions and consumables/obtained items. A menu button will also be present on the top right above the timer and score value in case the player wants to pause the game, or adjust certain settings, like perhaps audio volume for certain sounds.

**Screen Design (continued)**



**A more accurate size of the user interface would be the kind from Trap Adventure 2, I got the ideas for difficulty and moving obstacles from this game as well. If I used interface smaller than Super Mario’s games, and more like Trap Adventure 2, it could get the player to pay more attention to the game and less to outside distractions. Since if the user interface is too distracting, it might just affect their development of problem solving skills since they aren’t 100% concentrated on the game.**



**This is one more game I took inspiration from. In the tutorial for example, similar to this game; Dead Cells, some of the instructions could be in the form of speech bubble. If the instructions are physical and literally on screen, it doesn’t really teach the player anything since they are just tracing the game. Using speech and dialogue to tell players information, then getting them to do the actions themselves will progress their reasoning skills better than telling them exactly what to do and how. This can be used in conjunction with signs so there is a bit more variety in how the game’s tutorial interacts with the player.**

**Feedback** is vital to the development of your project. Present your ideas to the class and your teacher. Record below the comments that were made and how they have influenced the project development. What changes or improvements will you now be making in light of these comments?

Use this page for further notes on any of the sections above