

# Assignment Report

C PROGRAMMING FOR MSC

EXAM NUMBER: Y3863809

## Table of Contents

Requirements .....	2
Analysis .....	2
Specifications.....	3
Design .....	4
User Interface Design.....	4
Program Structure Design .....	6
Implementation Report.....	9
Testing and Verification.....	10
User Manual .....	10
Installation .....	10
System Requirements .....	10
Instructions.....	11
Obstacles .....	12
Additional Controls .....	12
Frequently Asked Questions.....	13
References .....	13

## Requirements

The project requires to develop a computer game using C with graphics lib which is a wrapper for allegro 5 and amio lib which is a wrapper for portaudio and portmidi libs. It shall be based on the game of golf with 9 different levels. The player controls the force of hitting the ball which is projected through a graphical golf course with obstacles. The aim is to hit the ball in to hole avoiding obstacles like tree, bunker and water. The flight of the ball must take basic physic like gravity and wind into account. If the ball hits an obstacle or leave the screen, player shall be penalized with extra stroke. If the ball leaves the screen it could be considered out of play and can start at the edge of the screen. A score shall be given considering the number of hits required and the difficulty of each levels. Total score shall be calculated based on the 9 levels of the game. Mouse shall be used to control the game and sound shall be added to make it more fun to play. Game shall also consider the audience with minimum computer literacy. Finally, it shall be fun and challenging to play with increasing difficulties.

## Analysis

Considering the requirements, Game shall start with a welcome menu where the player can choose to start a new game. The game shall have a player figure with a club hitting the ball, keyboard or mouse inputs could be used to set the angle of projection and the force with which the ball is hit. Graphics shall be drawn on a two-dimensional graphics window with some background music. Additional sounds can be added like the sound of club hitting the ball, ball bouncing, crowd cheering etc. Obstacles can have their own basic physics like tree can bounce the ball back considering the hitting angle, bunker/sand can slow down the ball faster than ground, water can drown the ball forcing the player to start again and ground shall stop the ball from falling infinitely because of the attraction of gravitational force. Also, to consider the wind, it can have dynamic wind behavior like changing direction and speed so that player face different type of challenge and can also make use of the effect on their favor. Clouds can be used to illustrate wind speed and the speed of wind shall have different level of impact on the ball and on the cloud considering their size and altitude.

Once the ball is hit, Newtonian formula of projectile can be used to calculate the location and speed of the ball.

$$\text{Horizontal Distance, } x = V_x t$$

$$\text{Horizontal Velocity, } V_x = V_{x0}$$

$$\text{Vertical Distance, } y = V_{y0} t - \frac{1}{2} g t^2$$

$$\text{Vertical Velocity, } V_y = V_{y0} - g t$$

Figure 1: Projectile Motion Formula (Projectile Motion Formula, n.d.)

In flight ball's trajectory is affected by other forces like wind speed and air friction. Wind speed could either increase or decrease the horizontal velocity depending on the direction where as air friction shall affect both horizontal and vertical speed trying to slow the projectile down.

Player may decide to leave the game and continue later. Therefore, an option to save progress and continue later would be helpful. Additionally, a leaderboard with ranking of player across game would motivate the player to score higher and play again.

As the game shall be designed for audience with minimum computer literacy, it shall have simple user interaction avoiding any complex key sequence. They shall not require any previous experience with computer game, neither shall they need any knowledge of actual golf game to enjoy this game.

## Specifications

The program specifications are given below

### 1. Core Features

- a. Create clickable buttons that can be placed at welcome screen
- b. Welcome screen with Continue, New Game, Leaderboard and Exit buttons
- c. Play background music throughout the game
- d. Draw a stick man with a golf club
- e. Animate the stick man when ball shall be hit
- f. Create ground that can be drawn, and it shall stop the ball on surface
- g. Draw an angle selection meter which can be controlled by mouse click
- h. Draw a speed selection meter that can be controlled by mouse
- i. Create tree that can be drawn and obstruct the ball
- j. Create pond that can be drawn on main game and drown the ball
- k. Create Hill that can be drawn and obstruct the ball
- l. Create bunker of sand that can slow down the ball much faster
- m. Create hole and draw it
- n. Create weather system to generates random weather pattern
- o. Build the level system where each level can be configured with ground, obstacles and hole
- p. Create the ball that can be shot with an angle and speed
- q. Draw ball trajectory
- r. Allow wind influence and air, ground and sand friction
- s. Allow bounce back from tree and Hills
- t. Add ball hit sound, Crowd cheering sound when ball falls in the hole and crowd boo sound when the ball is lost
- u. Pause screen with resume and exit level button

### 2. Additional Features

- a. Animated cloud that indicates wind speed
- b. Option for save and continue at the end of each level
- c. Game end screen to allow user to write their name
- d. Leaderboard to display the ranking of all players

## Design

To support most laptop and desktop screen sizes, game shall be developed for 1200x900 pixel window. This width and height of the window shall be stored in WIDTH and HEIGHT symbols respectively to allow easier modification of the screen size later. The game has five different screens welcome, play, pause, end and leaderboard. Each of these screens shall have their own input processing layers. The design for each of these layers are discussed below.

### User Interface Design

Program shall start with welcome screen and greet the user. It shall also provide options to start a new game, see previous rankings, continue from last saved state or exit. Therefore, the button names shall be

1. Continue
2. New Game
3. Leaderboard
4. Exit

Buttons shall be clickable and provide some visual feedback when mouse is hovered upon it. Continue button shall be hidden if there is no previously saved game. Flow diagram for the user interface is given below.

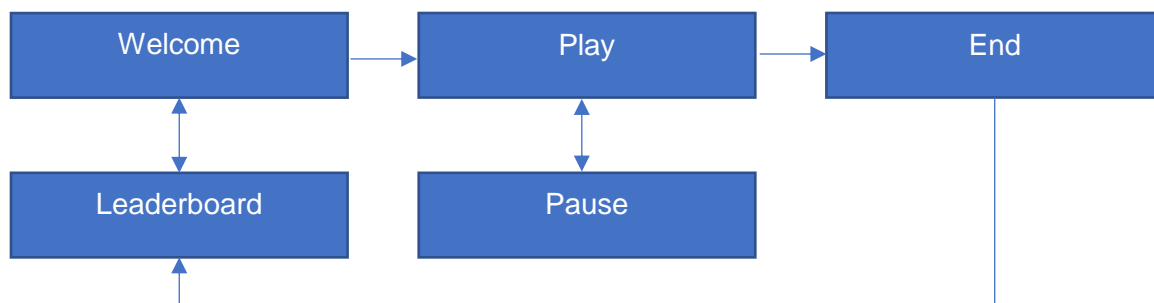


Figure 2: Flow diagramed for user interface

If the player decides to click 'New Game' button, user shall be taken to the play screen. On play screen, a graphical golf course with ground, stick man player, ball, hole and obstacles shall be drawn based on the level the player is playing. Play screen shall begin with an angle meter increasing and decreasing angle between 0 to 89 degrees. Player can left click anywhere on the screen to select the angle, as soon as the player clicks, angle on the meter shall stop at its current angle shall be considered as selected angle. Next the angle meter shall disappear, and a force meter shall appear which shall increase and decrease continuously between 0 to 100 and similarly

player can left click anywhere to select the desired force to hit the ball. Player can also right click anywhere to go back to angle selection mode. Once left mouse button is clicked on the speed selection mode, player shall animate a swing of golf club and the ball shall be considered hit with selected angle and speed. Ball shall visually display the trajectory it's taking, and it can hit an obstacle, land on ground, leave the screen or drop in the hole depending on its speed and angle properties. If ball is lost, level shall restart from beginning. If it lands on ground, player shall be moved close to ball and get the next chance to hit the ball again targeting the hole. Position of the initial ball position, initial player position and hole position shall be fixed on all games and shall be stored on following symbols.

Symbol Name	Value
<b>WIDTH</b>	1200
<b>HEIGHT</b>	900
<b>GROUND_LINE</b>	$(\text{HEIGHT} * 5) / 6$
<b>HOLE_X</b>	$(\text{WIDTH} * 9) / 10$
<b>HOLE_Y</b>	GROUND_LINE
<b>BALL_X</b>	WIDTH/6
<b>BALL_Y</b>	GROUND_LINE

Table 1: Basic symbols in use

Table 1 show the basic symbols in use where GROUND\_LINE is considered the game plain, ball shall not fall below this line.

Player can press ESC key on keyboard anytime on game screen to pause the game. On the pause menu player shall have 3 buttons Resume, Re-start Level and Exit Level respectively. Resume shall let the user to go back to the game, Re-start level shall allow user to reset hit counts for current level and start from the beginning and Exit level shall take the user back to welcome screen.

On play screen, once the ball lands on the hole, there shall be visual feedback congratulating the player, player animation can change to represent the win and user can left click anywhere in the screen to forward to the next level. At the end of each level progress shall be saved. Once the use complete level 9, user shall be forwarded to End Screen where the user can put their name on a Text box. This text box shall only allow keys A to Z. if the player hit enter key. End screen shall validate if the name is empty, if the name is not empty, end screen shall save the score for the player and take the player to the leaderboard to show ranking.

On the leaderboard, there shall be a list of names of previous players with their score on the right of their name, names shall be sorted in descending order according to their score. User can press ESC on keyboard of Right Click to go back to the welcome screen.

Someone can also click 'Leaderboard' on welcome screen to see the leaderboard with ranking of all previous player.

The final 'Exit' button on the welcome screen shall allow the user to quit the game.

## Program Structure Design

Program shall be well structured to allow easier update and maintenance; therefore, it shall be divided hierarchically into multiple modules and sub modules.

1. Golf
  - a. Game Menu
    - i. Button
  - b. Level
    - i. Player
      1. Animation
    - ii. Ball
    - iii. Ground
    - iv. Hole
    - v. Meter
    - vi. Tree
    - vii. Hill
    - viii. Lake
    - ix. Dune
    - x. Cloud
  - c. Weather
  - d. End Screen
  - e. Leaderboard
    - i. Game Score
  - f. Sound Player

Each of the modules above are sub divided into two different sub modules, paint and update. The purpose of paint is to draw on the screen where as update deals with game logics like environment, animation, weather, collision with obstacle, ball drop in the hole etc.

At the beginning, main function shall initialize game resources, load images, load sounds, create the graphics window, start audio system and then start the main game loop.

Main loop of the game shall be implemented using a finite state machine with 5 different states

1. Welcome
2. Play
3. Pause
4. End
5. Leaderboard

State flow from one state to another and direction is illustrated on figure 2.

On the welcome screen, only accepted input is left button click, pseudo code to check hover and click on the button is given bellow

```

function check_mouse takes a pointer to button structure as input
    if the mouse x is bigger than button x and smaller then button x + button width
        if the mouse y is bigger than button y and smaller than button y + button height
            if mouse left button down
                return button click
            end if
        return button hover
    end if
end if
return none
end function

```

Code 1: Pseudocode to determine if mouse was clicked on a button

If the button is hovered, back colour of the button is changed to highlight the hover status. If clicked each button only change the state of the main state machine

Button Name	Actions
<b>Continue</b>	Set State to GAME_PLAY
<b>New Game</b>	Set State to GAME_PLAY Current Level = 0
<b>Leaderboard</b>	Set state to GAME_LEADERBOARD
<b>Exit</b>	Set to run flag to 0 signaling game to close

Table 2: Corresponding action for welcome screen buttons

Play screen shall have its own internal finite state machine with 4 different states, keeping track of the modes and actions the user can take on any mode. Logical flow of internal modes is given in figure 3. On Select Angle mode, player animation at initial state, ball is static, angle meter is visible and force meter is hidden. User can only click left mouse button to confirm the angle and move on to select speed mode. On select speed mode, angle meter is hidden, and force meter is visible. Use can left click to hit the ball or right click to go back to select angle mode.

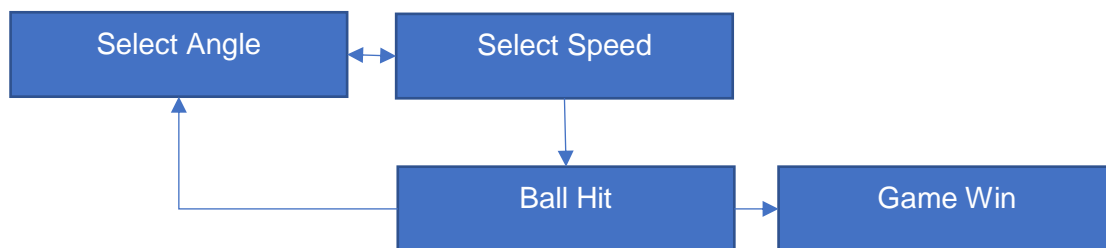


Figure 3: Logical flow of state on play screen

On ball hit mode, player animation starts playing and stop at the end of the animation, play golf club swing sound and ball is projected with previously selected angle and speed. Hit count for current level is increased, if the ball is lost, ball and player is moved to initial position otherwise if

the ball is on ground or sand, player is moved to ball and game mode is switched to select angle. If the ball drops in the hole, game mode is switched to game win. On game win screen, player can only left click to continue to next level or to the end screen.

Player animation is created based on a state machine with multiple images, when the player animation is played, frames are changed to each tick to produce the animation.

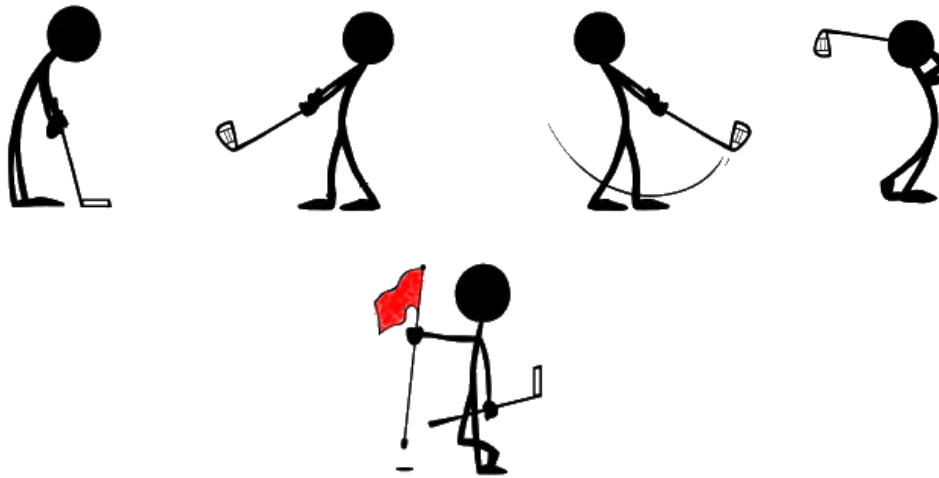


Figure 4: Animation Frames (Bridson, 2015)

Each obstacle shall be created using three geometrical shapes, circle, rectangle and triangle to make obstacle detection easier. For example, a rectangle can be used to create a tree trunk and circles can be used to form the leaves. Using basic geometric shapes allows to check if the balls center lay within these shapes. Therefore, hit on obstacle can be calculated based on where the ball center is within any of the shapes used as obstacle. Collation with rectangle is determined by a similar algorithm to Code 1. Collision detection with a circle is develop based on the distance from the center of the circle.

```
function hit_circle takes a circle and point as input
    calculate distance from point to center of circle and set it to distance
    if distance is greater than circle radius then
        return collision
    end if
    return no collision
end if
```

Code 2: Pseudo code to determine if there was a collision with a circle

Finally, collision with triangle is determined based on the sign of cross products (kingnosis, 2005).

Ball also has its own internal states, BALL\_STATIC, BALL\_IN\_FLIGHT, BALL\_IN\_FLIGHT, BALL\_ON\_SAND, BALL\_IN\_HOLE and BALL\_LOST. On each tick ball's current horizontal and vertical position and speed are updated. For each obstacle and hole obstacle collision is checked and balls internal state is updated accordingly.

Environmental factors like gravitational force, air friction, ground friction, sand friction etc. Shall be store in symbols to allow easy modification.

Maximum score for each level is calculated based on a base point sored on BASE\_POINT\_PER\_LEVEL symbol where

$$\text{Max Point} = \text{BASE\_POINT\_PER\_LEVEL} \times \text{Level Number}$$

Score at the end of each level is calculated based on the number of hit the user took to finish the level,

$$\text{Score} = \text{Max Points} - (\text{HIT\_LOSS\_CONSTANT} * \text{Number of hits})^2$$

A minimum score threshold is set by MINIMUM\_POINT symbol which defines the minimum percentage of max score the user shall be awarded.

## Implementation Report

Project is implemented according to the design, keeping each module on a separate header and source file. The main function of the program is in *assignment.c* file which includes the golf module and run it inside the loop. All game symbols are present in *common.h* except the status symbols used by individual modules.

Main function creates an instance of GOLF structure. Then it initializes the instance and start the main loop. Two main functions are called inside the loop, *golf\_update* and *golf\_paint*. *golf\_update* runs the main state machine to switch between welcome, play, pause, end and leaderboard. It also calls respective update function for other modules. It returns 0 once the exit button on welcome screen is clicked, otherwise it returns 1. *golf\_paint* on the other hand is solely responsible for drawing the graphical golf course, welcome screen, menus, buttons etc.

Finally, when *golf\_update* returns 0, main loop breaks and call *golf\_destroy* which deallocate the memories and close graphical window and sound system.

Some of the modules like GAMEBUTTON, ANIMATIONCLIP, BALL, GAMELEVEL and ENDSCREEN use callback functions to interact with GOLF. Through out the game there is only two global variables, *golf\_game* which is a pointer to a GOLF instance, accessed by the callback functions from other modules to update status or report completion. The second global variable is *\_weather* which is an instance of GAMEWEATHER, the purpose of this global variable is to maintain weather consistency between update and paint function calls. None of these global variables are accessed from outside the modules where they are defined.

Some modification on the original *graphics\_lib.h* and *graphics\_lib\_functions.c* has been carried out to support some additional colours used by the game.

All the core and additional features has been achieved and successfully implemented in the game.

## Testing and Verification

As the program is developed in a modular fashion, each of the smallest components are developed independently and tested rigorously before integrating with a bigger module. This bigger module is then tested further to point out any issue or inconsistencies. Therefore, the software was tested both on its smaller unit level and on the functional level.

User testing has also been carried out to spot additional bugs and to determine if the game is fun to play. There were some inconsistencies with initial design like the assumption of gravitational constant was too high causing the ball to fall up faster. In the implementation instead of using projectile equations directly, ball position and velocity is updated on every tick and any previous information about initial state is removed. Other environmental constants like air friction, wind influence on ball, wind influence on cloud, ground friction, sand friction, maximum hit speed variables are also optimized based on the test results.

Each game levels are also tested individually to make sure they are achievable and not to frustrating to play while posing enough challenge for the player. Therefore, the position and size constants of the objects are also updated accordingly.

After several user testing was confirmed that the game meets the requirements provide and satisfies the planned specifications.

## User Manual

Instructions for installation, system requirements and gameplay instructions are given on respective sections below.

### Installation

Installation of the game only requires coping the executable golf.exe along with *data* folder and following libraries

libportaudio-2.dll

msvcr100d.dll

portmidi.dll

All the above-mentioned files and folders shall remain in the same directory.

### System Requirements

Golf game is a lite weight game developed using C which doesn't require much computing resources. Its usages a maximum of 40MB of RAM. Therefore, any computer that can run a Windows 10 operating system shall be able to run this game. Considering the criteria, minimum requirements for running the game is given bellow (Microsoft, 2017).

Operating System	Windows 10
CPU	1 GHz
RAM	1GB
Hard disk space	16GB
Graphics card	DirectX 9 or later with WDDM 1.0 driver
Display	1200x900

## Instructions

Game can be started by double clicking on the golf.exe executable. User is greeted with a welcome screen like below



Figure 5: Welcome Screen

Please click on 'New Game' to start from level 1 or click 'Continue' to resume from the last saved state.

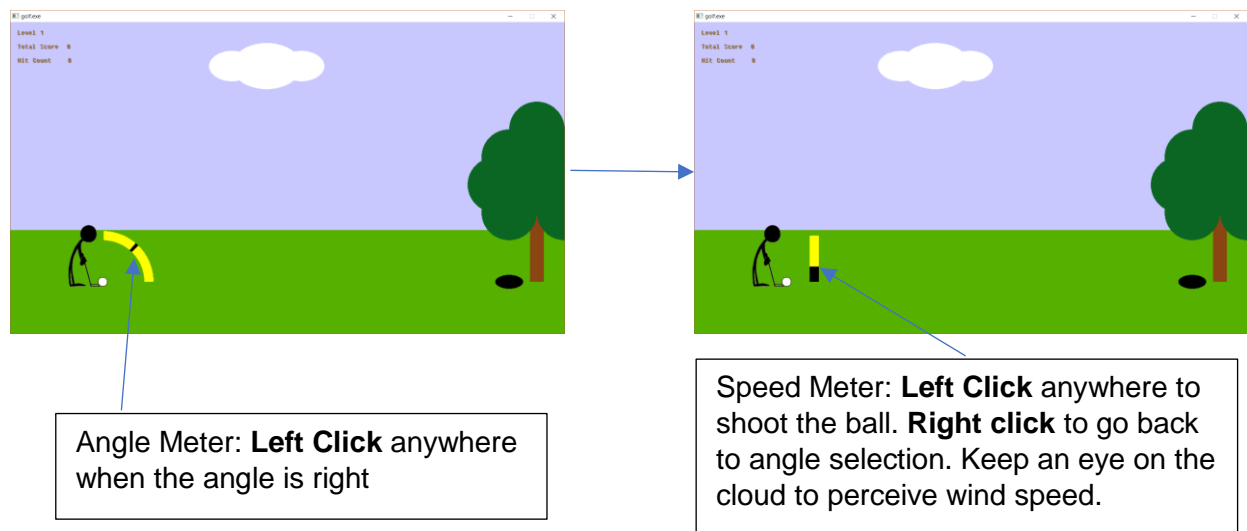



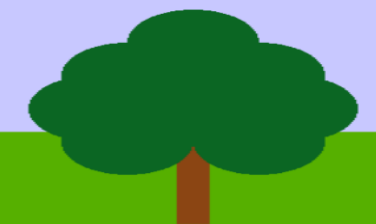


Figure 6: Angle and Speed Selection

Level number, total score and hit count is displayed on the top left corner of the screen. Ball need to be dropped in the black hole at the end of the screen to complete the level.

### Obstacles

Game has 4 different type of obstacles, each of those are explained below

	<p>Sand dune, it has higher friction than ground and air, ball will slow down immediately and will not bounce when land on it. Hit the ball with greater force if you are hitting the ball that is on a sand dune.</p>
	<p>Lake, ball will drown if it lands in the lake, level starts from beginning when the ball lands in a lake or leave the screen. Try to avoid landing on them, please shoot with a higher angle to go past the lake.</p>
	<p>Hills, ball would bounce back when it hits the hill, if the ball is slow when it hits the hill, it might get lost. In that case game will start from beginning.</p>
	<p>Tree, ball will bounce back when it hits the tree, some trees can be bigger and hard to avoid. On some levels, trees can be a blessing stopping the ball from leaving the screen or bouncing it back to the hole.</p>

### Additional Controls

During a game ESC can be pressed on the keyboard to pause the game. Please pause the game if you want to re-start or exit the level.

After level 9 is complete, a text box will appear asking for name, please type in the name and press enter to confirm. Once completed, leaderboard will appear with the list of previous scores. Leaderboard can be closed by pressing ESC or Right clicking anywhere on the screen.

## Frequently Asked Questions

**Question:** I could shoot through a tree.

**Answer:** There is a known bug, if you shoot too close from a tree trunk, you can shoot through it if the ball speed is greater than the width of the trunk.

**Question:** Why my ball is lost in the hills?

**Answer:** Ball can be lost in the hills if the horizontal speed of the ball is less than 3, in that case ball will not return from the hills.

**Question:** I shoot with high force, but the ball doesn't go far.

**Answer:** Please keep an eye on the clouds, if the clouds are moving on the opposite direction, ball will be affected by the wind, therefore it might decelerate faster and land closer than expected.

**Question:** My ball landed further right of the hole, how can I shoot it back to the hole?

**Answer:** Now ball can not be shot from right to left, only option is to shoot it normally so that it leaves the screen and ball is returned to original position or restart the level from pause menu.

**Question:** Why the game animation is lagging?

**Answer:** It can happen, specially when the laptop is not plugged in, because the clock speed of the processor would be restricted making the game slower.

## References

Bridson, T. (2015, 08 19). *Set of stick figures playing golf– stock illustration* . Retrieved from DepositPhotos: <https://depositphotos.com/81134806/stock-illustration-set-of-stick-figures-playing.html>

kingnosis. (2005, 01 21). *Is this a better point in triangle test (2D)?* Retrieved from gamedev.net: <https://www.gamedev.net/forums/topic/295943-is-this-a-better-point-in-triangle-test-2d/>

Microsoft. (2017, 11 20). *Windows 10 system requirements*. Retrieved from Windows Support: <https://support.microsoft.com/en-ie/help/4028142/windows-windows-10-system-requirements>

*Projectile Motion Formula*. (n.d.). Retrieved from Tutor Vista: <https://formulas.tutorvista.com/physics/projectile-motion-formula.html>