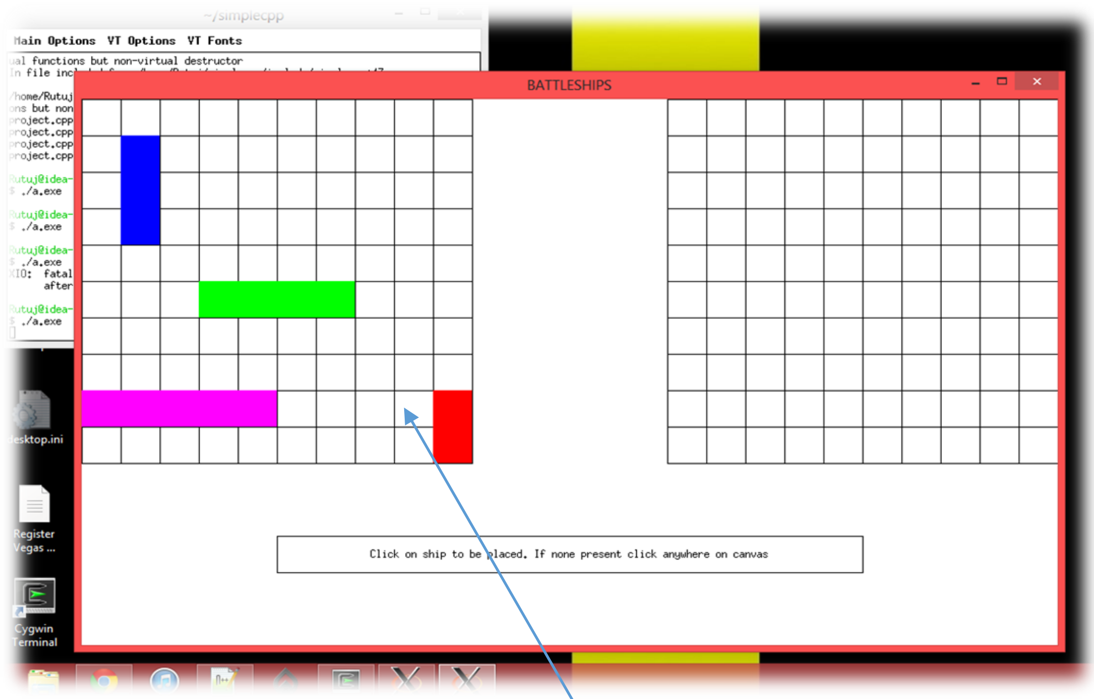
Vinayak Agarwal and Rutuj Shah
13D100023 and 13D100022

CS101 Project:BattleShips

Objective: Effective utilisation of c++ programming and simplecpp graphic skills by recreating a classic board game into its playable , bilateral computer counterpart.

Aim: It is a two player game each controlling a fleet of ships. Each player has to place his/her own ships and locate the other's, each on an individual battlefield. The player who locates all the ships of the enemy before the enemy does the same to him/her wins the game. The result of both the events occurring simultaneously is a draw.

Input: Each player has the power to place his/her ships(4 in number) on his/her battlefield depicted by a grid. The player has to select the type of ship from the interface and click on the starting and the ending cell of the ship to be placed. After placing all the ships Player two has to do the same.

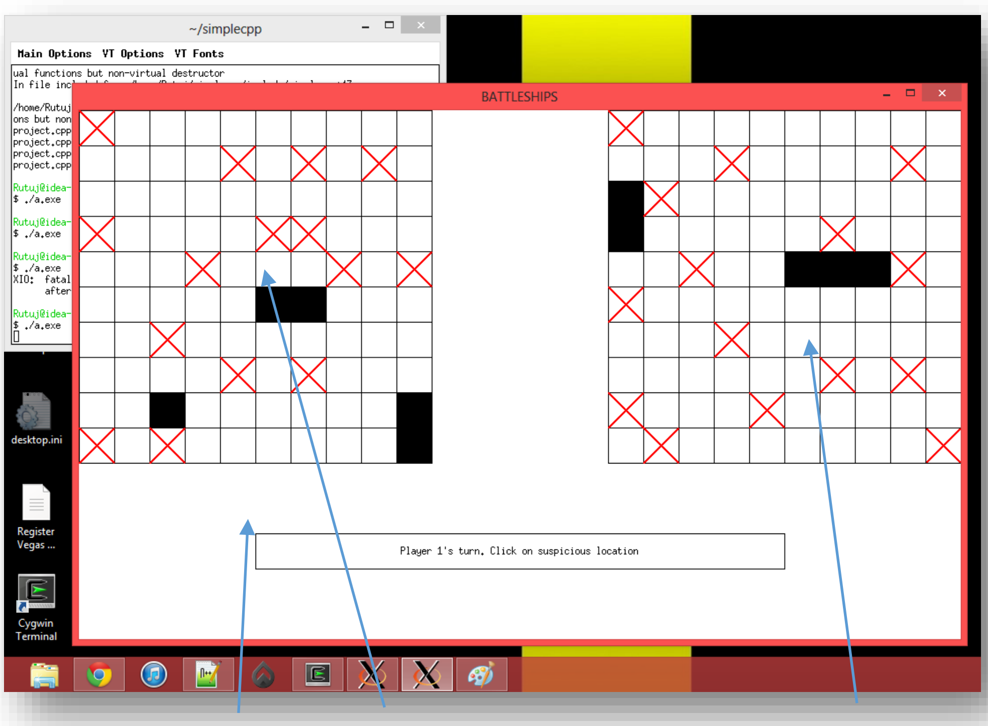


```
ual functions but non-virtual destructor
In file inc

/home/Rutuj
ons but non
project.cpp
project.cpp
project.cpp
project.cpp

Rutuj@idea-
$ ./a.exe

Rutuj@idea-
$ ./a.exe

Rutuj@idea-
$ ./a.exe
XIO:  fatal
      after

Rutuj@idea-
$ ./a.exe
```

```
Click on ship to be placed. If none present click anywhere on canvas
```

All the ships being placed

Output: The placement of ships would colour the cells of the grid containing the ship. During Gameplay, Each missed hit would correspond to a cross being displayed on that cell and each successful shot would blacken the cell.



```
ual functions but non-virtual destructor
In file inc

/home/Rutuj
ons but non
project.cpp
project.cpp
project.cpp
project.cpp

Rutuj@idea-
$ ./a.exe

Rutuj@idea-
$ ./a.exe

Rutuj@idea-
$ ./a.exe
XIO:  fatal
      after

Rutuj@idea-
$ ./a.exe
```

```
Player 1's turn. Click on suspicious location
```

Instruction Panel Successful hit Missed hit

After one player locates the ships of the other the instruction panel declares the win of the player and then shows the location of the ships to both the players and the game ends.

Algorithm:

START

1. Select ship.
2. Click on initial and final position of ship on the grid.
3. If all ships are placed on the grid go to step 6 else continue

4. If clicks lie on Player's grid and clicks represent valid positions of ships such that the ships are of selected type and they don't overlap, go to step 5.
   Else go to step 1.
5. Store the selected positions for a certain ship in an array
   Display the selected ship on the grid and go back to step 1.
6. Hide the placed ships.
7. Repeat steps 1-6 for player 2.
8. Player 1 clicks on a certain suspicious position on the grid.
   If the guess is successful display the part of the ship clicked on else display a cross on the cell which represents a failed guess
9. Player 2 does the same thing as Player 1 in step 8.
10.
   (a) If all of player 1's ships are detected player 2 wins. End.
   (b) If all of player 2's ships are detected player 1 wins. End.
   (c) If player 1's ships are detected at the same instance as player 2's ships the game the game is a draw. End.
   (d) If none of the conditions a,b,c hold true then goto step 8.

STOP

<u>Major functions:</u>

- void Grid(Rectangle &r) - Function to create the grid.
- void color(double x1,double x2,double y1,double y2,int redval,int blueval,int greenval) - Function colours the ship with end coordinates (x1,y1) -> (x2,y2) with colour attributes COLOR(redval,blueval,greenval).
- int xCoord(int clickVal) - Converts x coordinate of click to respective position in grid. E.g. 42 = 1, 348 = 8
- int yCoord(int clickVal) - Converts y coordinate of click to respective position in grid.
- bool checkVal(int x,int y,int o) - Checks whether or not click lies in grid.

- void setup(int choice) - Function for setting up the ships for selection.
- int shipType(int ship) - Returns the type of ship according to size.
- bool checkOverlap(int x1[],int x2[],int y1[],int y2[],int val[],int ch) - Checks whether ship to be placed overlaps or not and also that 1x1 ships are disallowed..
- void create(int x1[], int y1[], int x2[],int y2[],int o) - Setting up ship positions.
- void hide(Rectangle &r,int o) - Hides ships after input from canvas.
- void drawCross(int x,int y) - Draws crosses on grid for missed chances.

<u>How To Run The Code:</u>

The procedure is well explained under the *Input/Output* and the *Algorithm* subsections. Also, onscreen instructions are provided while running the program.

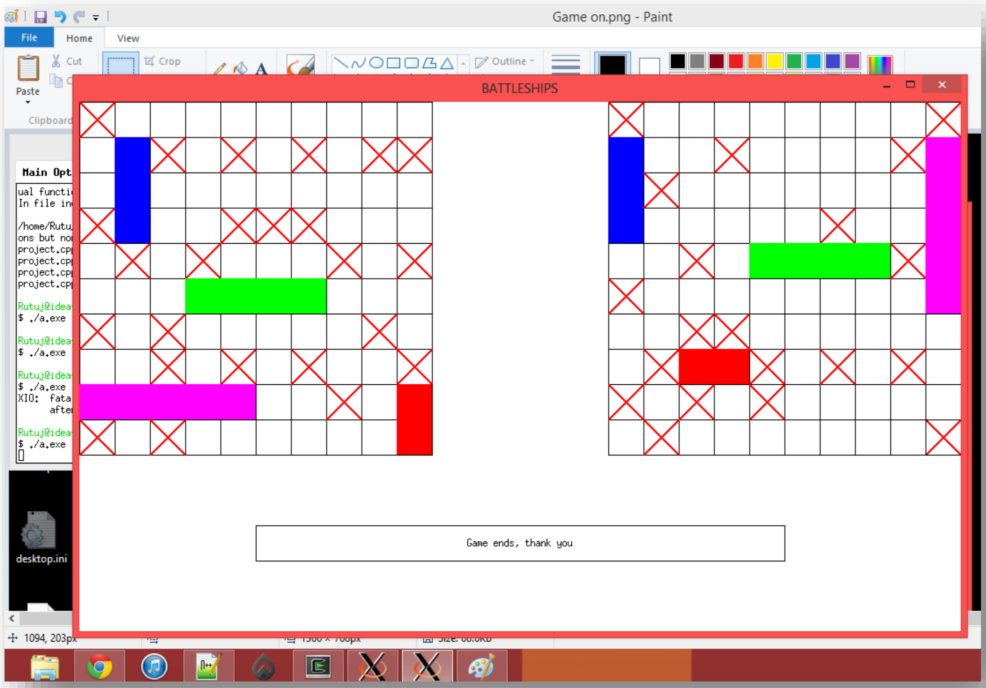
<u>Special Features Of The Program</u>

Specific conditions to eliminate the following bugs

- Ships of dimensions 1x1 are placed.
- Ship placed is not of the selected type.
- Ship overlap.

Implementing programming tricks like the following to shorten the program

- Writing code for different functions so that its generalized for both the grids by using the fact that both the grids have different origins.
- Using a single function to colour different ships.



```
Main Opt
ual functi
In file in

/home/Rutu
ons but no
project.cp
project.cp
project.cp
project.cp

Rutuj@idea
$ ./a.exe

Rutuj@idea
$ ./a.exe

Rutuj@idea
$ ./a.exe
XIO:  fata
      afte

Rutuj@idea
$ ./a.exe
```

Game ends, thank you