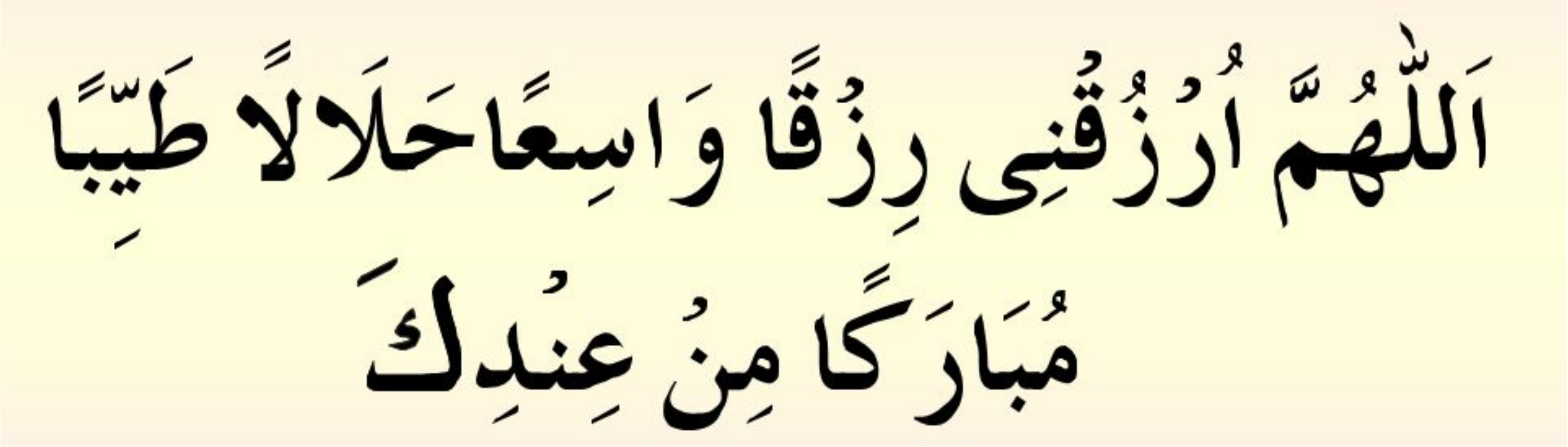
Display the output on Console at a Specific Location

اَللّٰهُمَّ اُرُزُقُنِى عِلُمًا نَافِعًا وَاسِعًا عَمِيُقًا

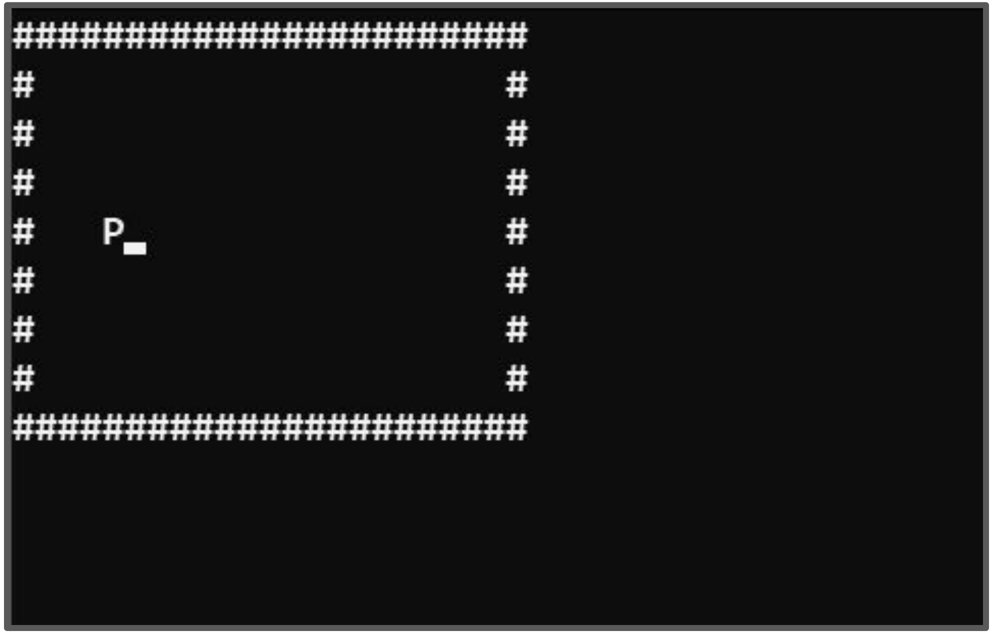


اَللّٰهُمَّ اُرُزُقُنِى رِزُقًا وَاسِعًا حَلَالًا طَيِّبًا

مُبَارَكًا مِنُ عِنُدِكَ

Goal: Move Player on the Console

The goal of this lecture is to move player on the console horizontally.



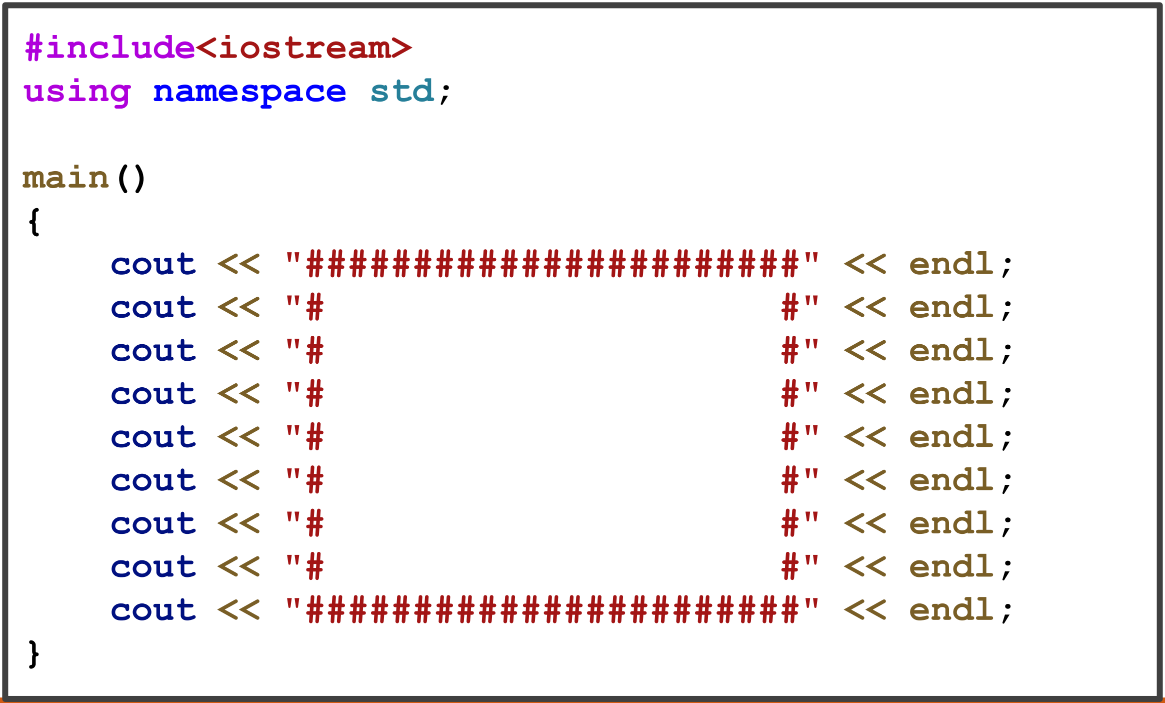
```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################
```

Step 1: Print maze on the Console

Lets print the maze on the console.

Step 1: Print maze on the Console

Lets print the maze on the console.

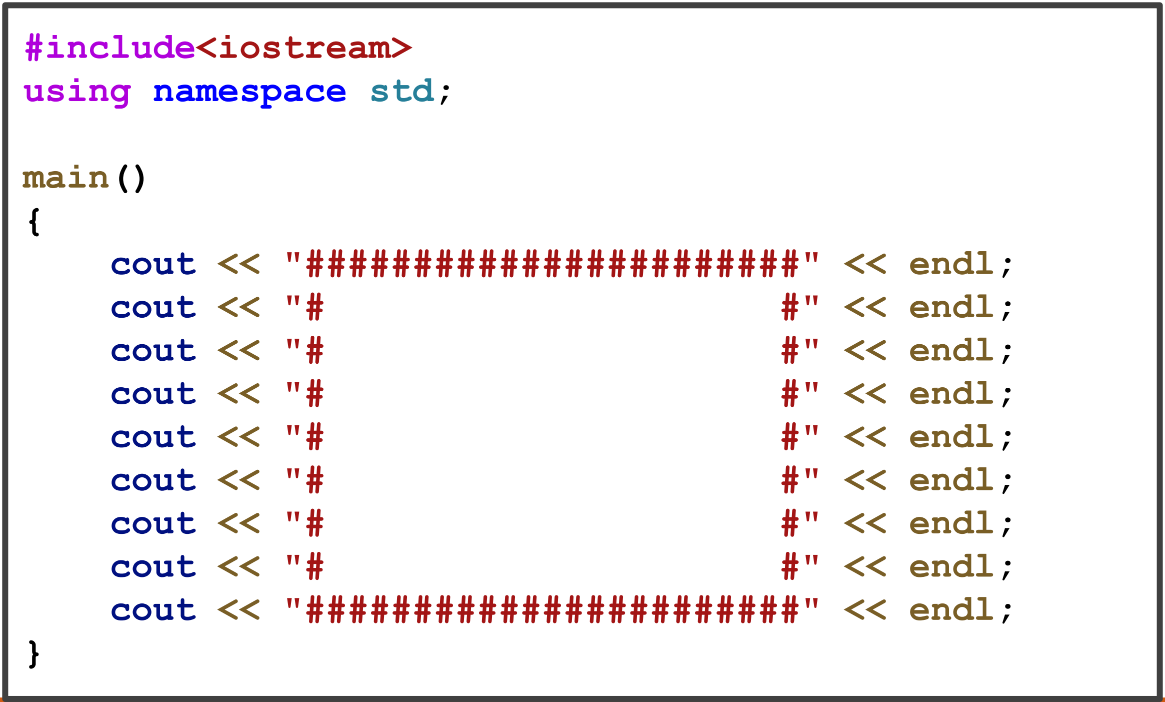


```cpp
#include<iostream>
using namespace std;

main()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 1: Print maze on the Console

Can we make this code better?



```
#include<iostream>
using namespace std;

main()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 1: Print maze on the Console

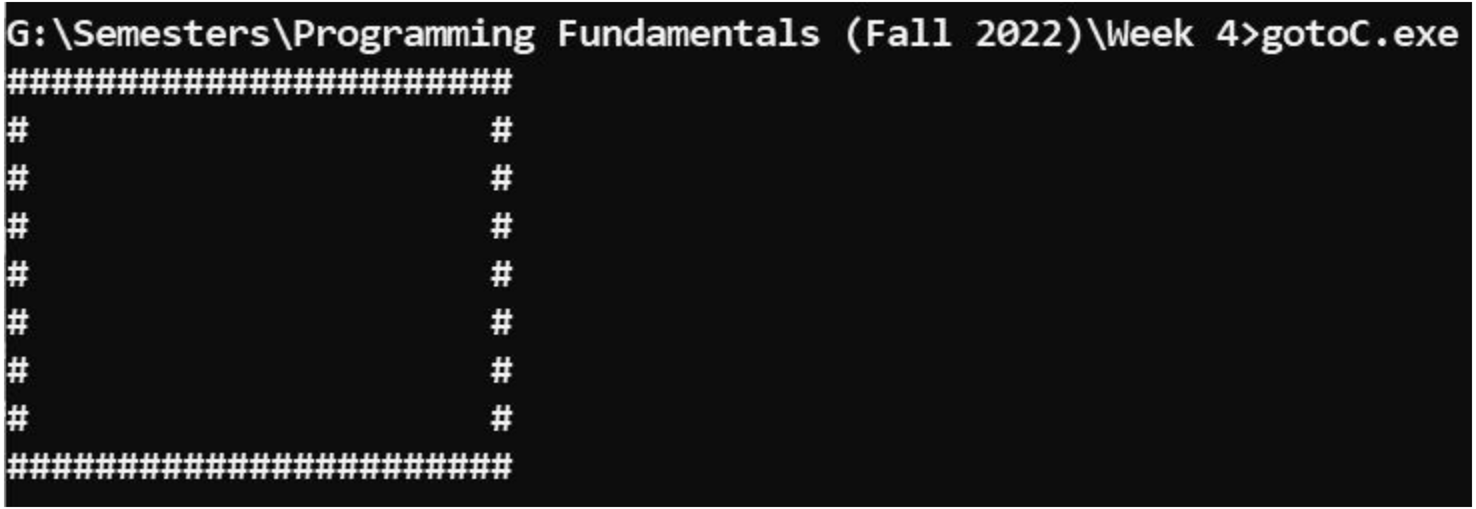
Yes, we can make a function to print the maze.

```
#include<iostream>
using namespace std;
void printMaze();
main()
{
    printMaze();
}

void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 1: Print maze on the Console

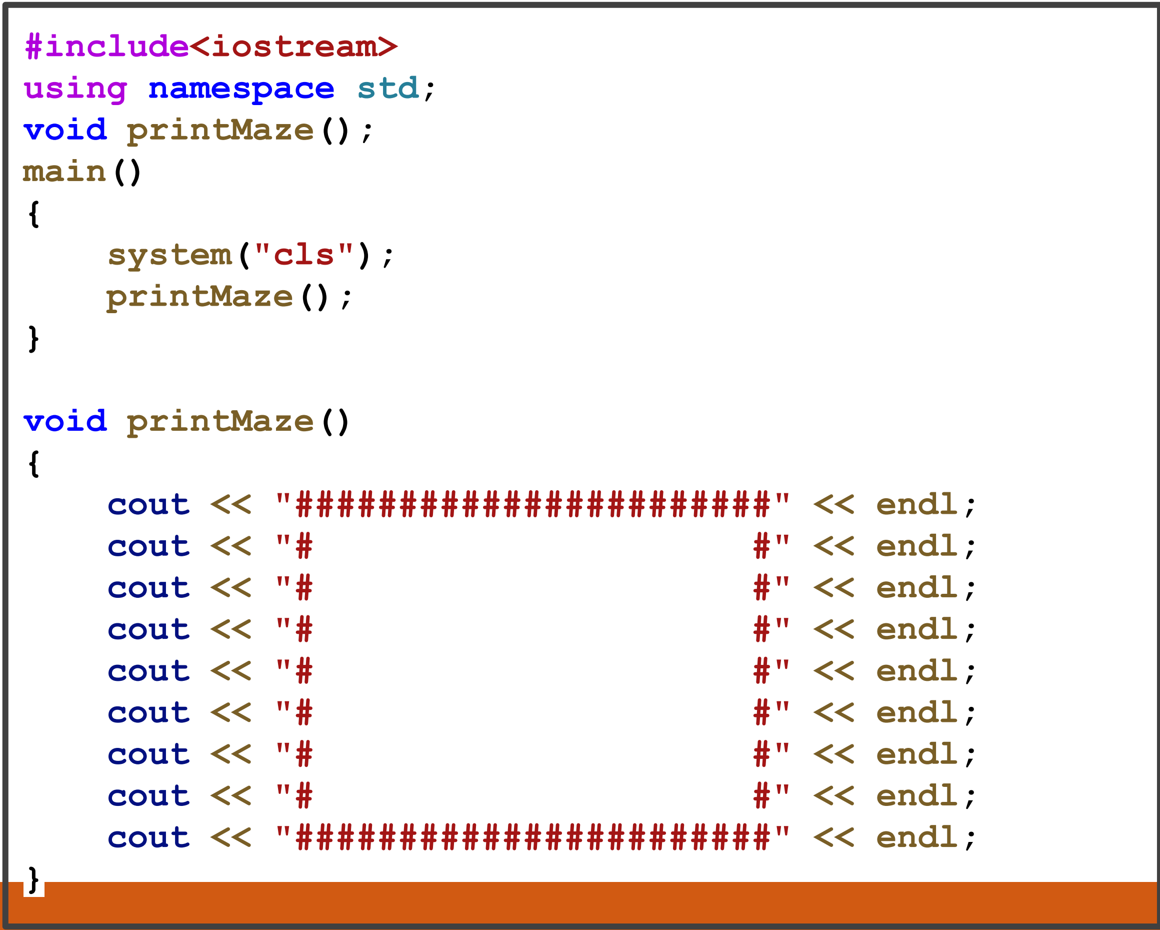
Console is also showing the directory path.
What if we want to remove that and we just want to print the maze on the console?



```
G:\Semesters\Programming Fundamentals (Fall 2022)\Week 4>gotoC.exe
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
```

Step 1: Print maze on the Console

We can **system("cls")** command before printing the maze.



```
#include<iostream>
using namespace std;
void printMaze();
main()
{
    system("cls");
    printMaze();
}

void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

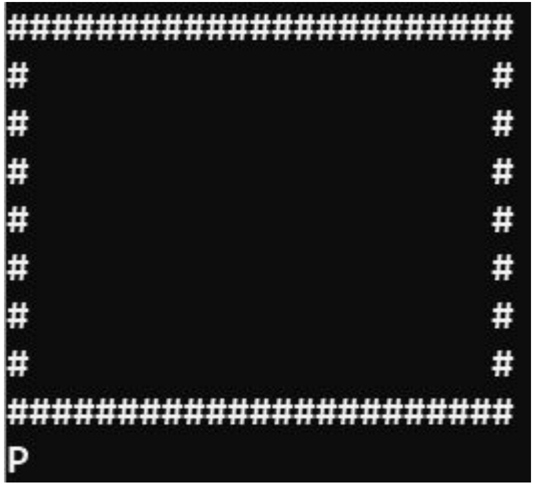
Step 2: Print the player on Console

After printing maze, we can print the player on the console.

```
#include<iostream>
using namespace std;
void printMaze();
main()
{
    system("cls");
    printMaze();
    cout << "P";
}
void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 2: Print the player on Console

Output

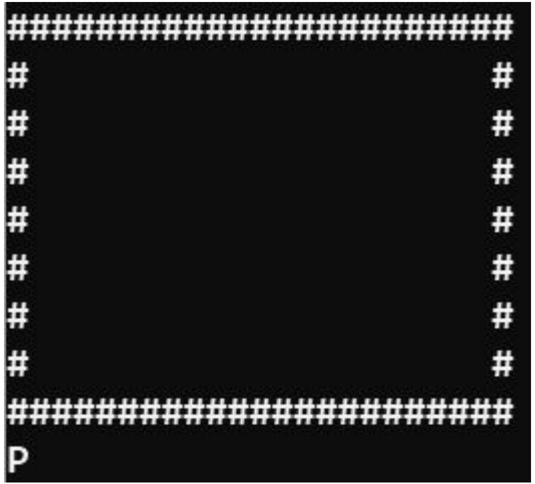


```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

```
#include<iostream>
using namespace std;
void printMaze();
main()
{
    system("cls");
    printMaze();
    cout << "P";
}
void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 2: Print the player on Console

But we want to print P on the specific location inside the maze.

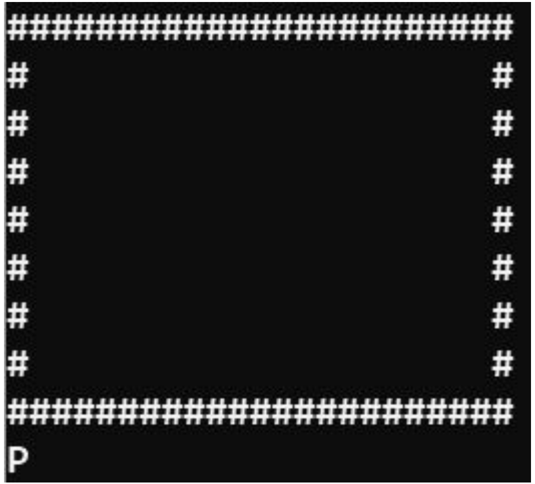


```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

```
#include<iostream>
using namespace std;
void printMaze();
main()
{
    system("cls");
    printMaze();
    cout << "P";
}
void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 2: Print the player on Console

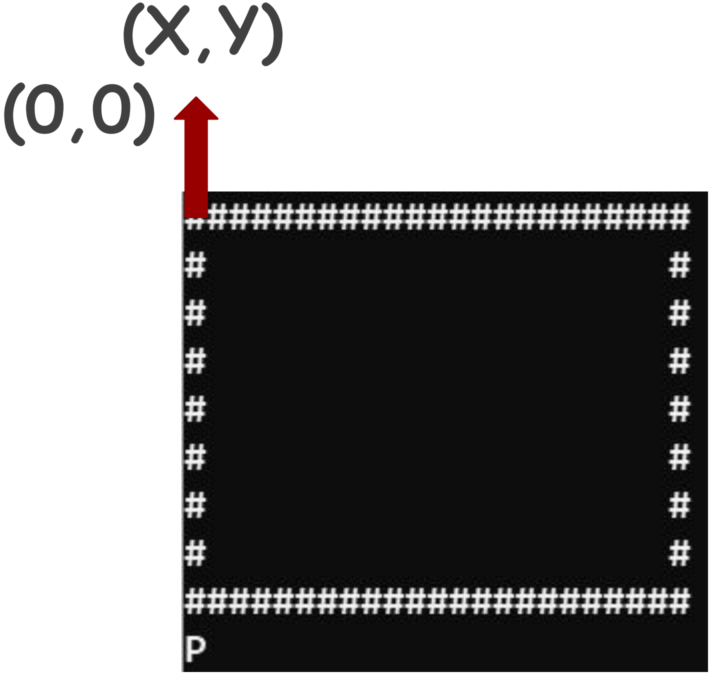
For that lets first see how console is divided into X and Y coordinates



```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

```
#include<iostream>
using namespace std;
void printMaze();
main()
{
    system("cls");
    printMaze();
    cout << "P";
}
void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

X and Y Coordinates of Console

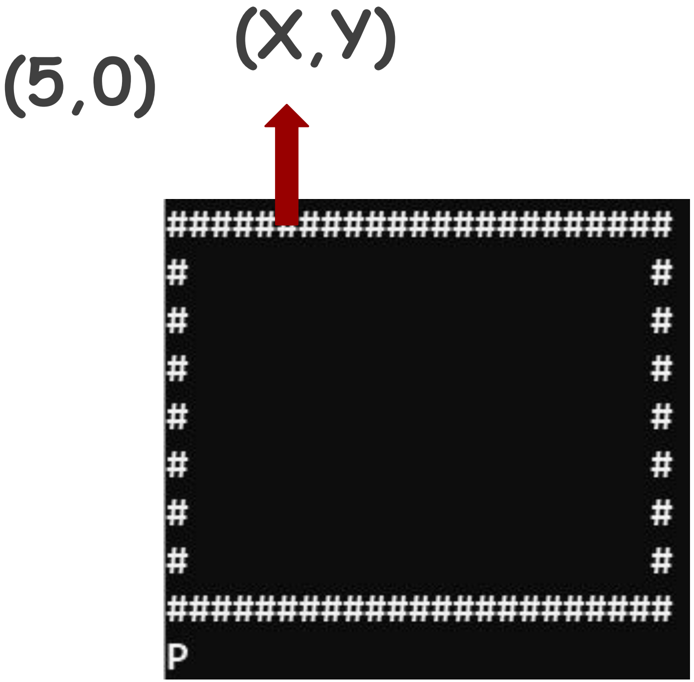


(X,Y)

(0,0)

```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

X and Y Coordinates of Console

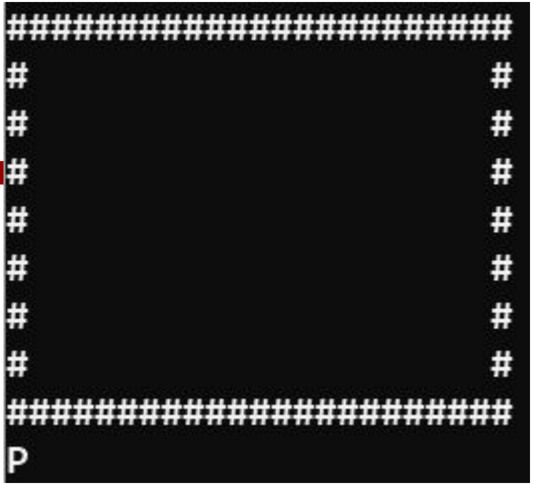


```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

X and Y Coordinates of Console

(X,Y)

(0,3)

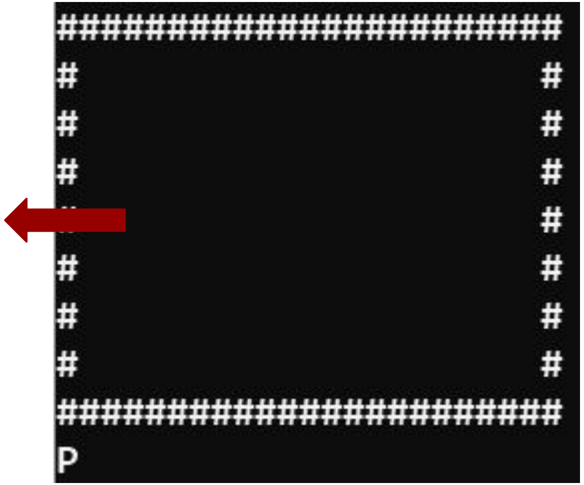


```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

X and Y Coordinates of Console

(X,Y)

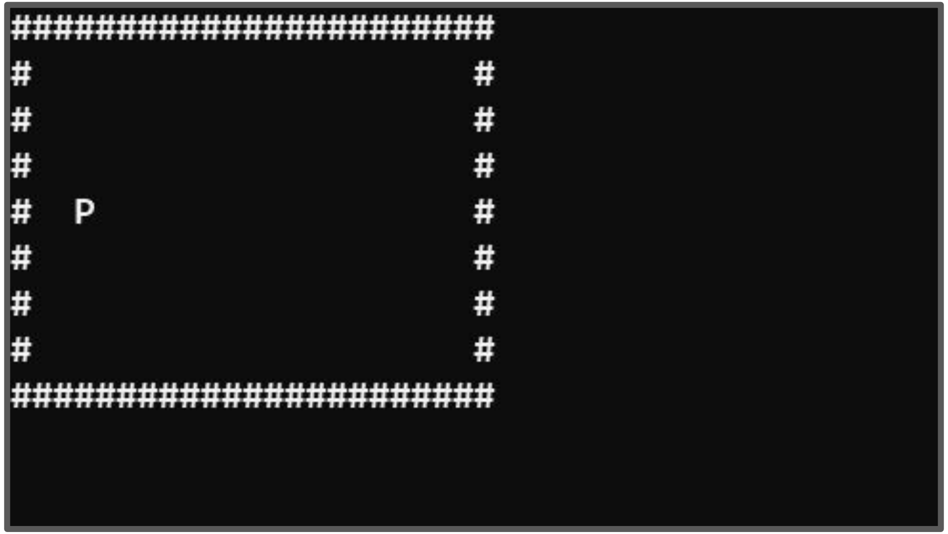
(3,4)



```
#######################
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#                     #
#######################
P
```

Place the cursor on specific location

We will use a function gotoxy() and we will pass it the x coordinates of the console and y coordinates of the console and it will place the cursor on specific location on console.



```
#######################
#                     #
#                     #
#                     #
#  P                  #
#                     #
#                     #
#                     #
#######################
```

gotoxy() Function

To use gotoxy() function we have to include windows.h file.

```
#include <windows.h>
```

gotoxy() Function

The definition of gotoxy() function is given by:

```
void gotoxy(int x, int y)
{
 COORD coordinates;
 coordinates.X = x;
 coordinates.Y = y;
 SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coordinates);
}
```

gotoxy() Function

We will not go into the functionality of this function, we will just copy this function in our program and use it.

```
void gotoxy(int x, int y)
{
 COORD coordinates;
 coordinates.X = x;
 coordinates.Y = y;
 SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coordinates);
}
```

Step 2: Print the player on Console

```
void gotoxy(int x, int y)
{
 COORD coordinates;
 coordinates.X = x;
 coordinates.Y = y;
 SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coordinates);
}
void printMaze()
{
    cout << "#######################" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#                     #" << endl;
    cout << "#######################" << endl;
}
```

Step 2: Print the player on Console

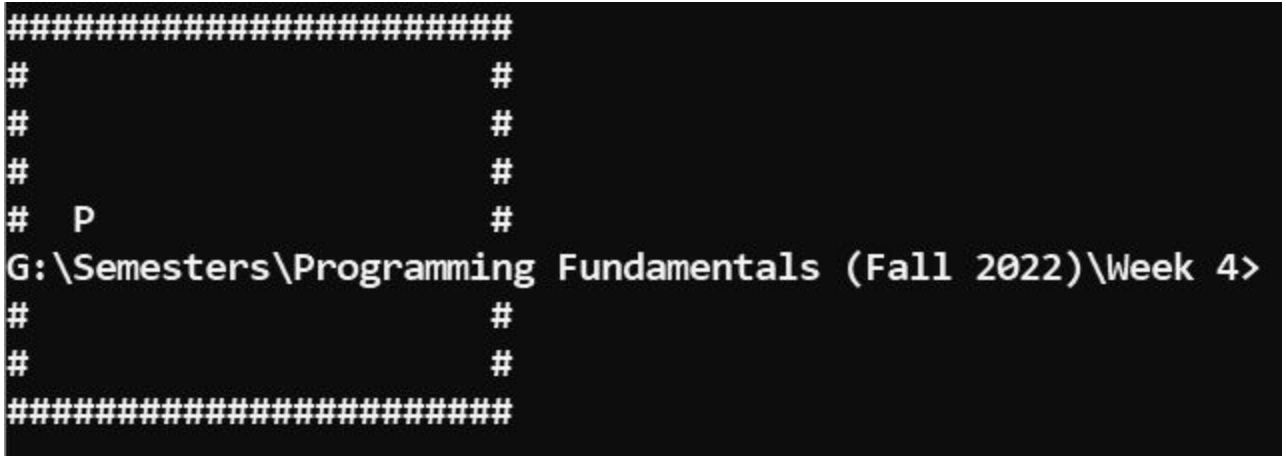
```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
        system("cls");
        printMaze();
        gotoxy(3, 4);
        cout << "P";
}
```

Step 2: Print the player on Console

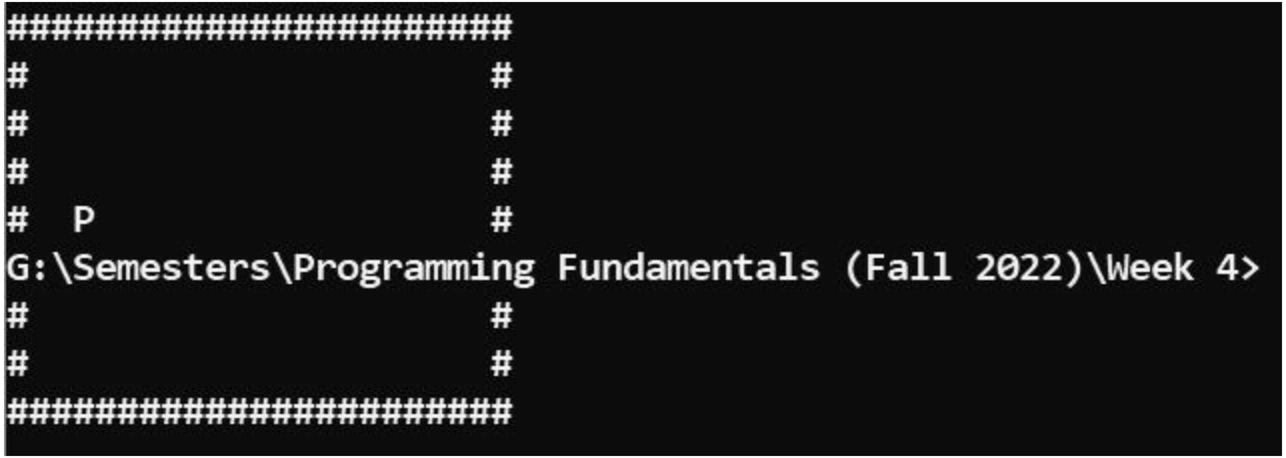
Now we have printed the Player on the specific location.



```
#######################
#                     #
#                     #
#                     #
#  P                  #
G:\Semesters\Programming Fundamentals (Fall 2022)\Week 4>
#                     #
#                     #
#######################
```

Step 2: Print the player on Console

Why this **prompt line** is printing here ?

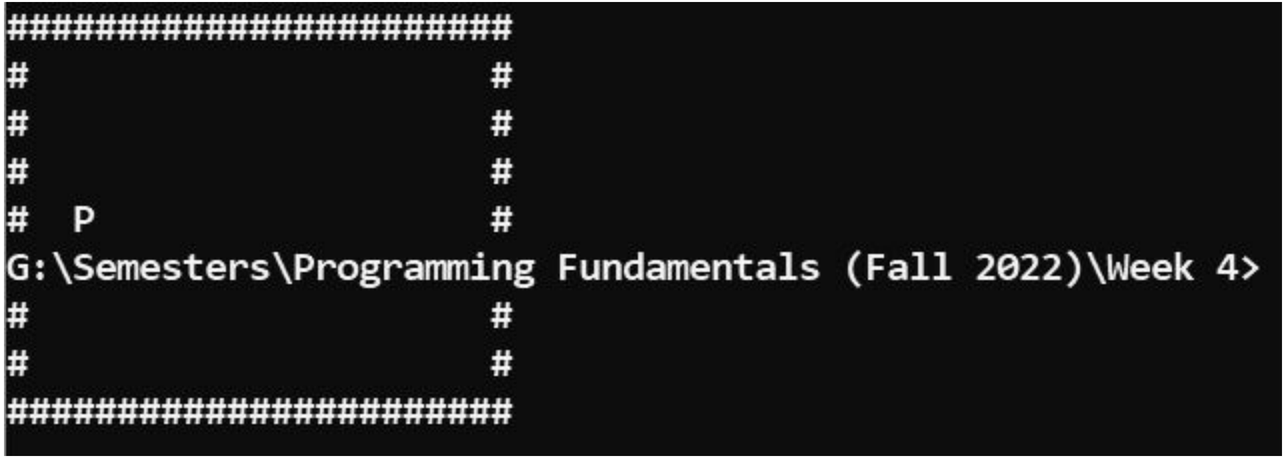


```
########################
#                      #
#                      #
#                      #
#  P                   #
G:\Semesters\Programming Fundamentals (Fall 2022)\Week 4>
#                      #
#                      #
########################
```

Step 2: Print the player on Console

Why this prompt line is printing here ?

It is printing after last location of gotoxy. So if we want to print it after the grid, therefore, we need to add extra gotoxy at the end of the code



```
#######################
#                     #
#                     #
#                     #
#  P                  #
G:\Semesters\Programming Fundamentals (Fall 2022)\Week 4>
#                     #
#                     #
#######################
```

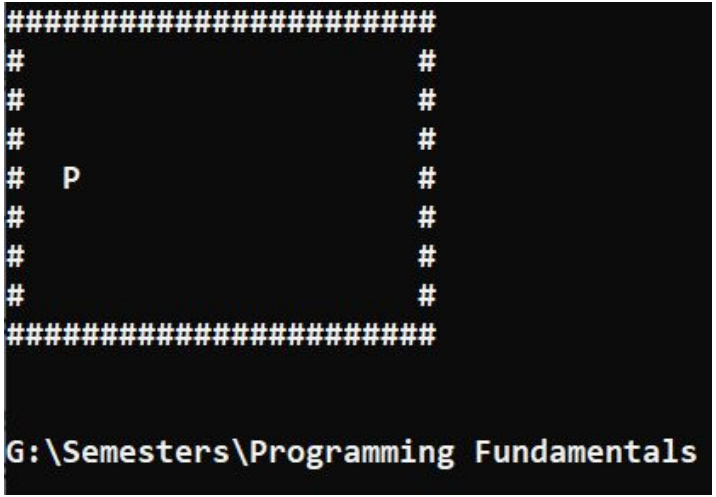
Step 2: Print the player on Console

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
        system("cls");
        printMaze();
        gotoxy(3, 4);
        cout << "P";
         gotoxy(0, 10);

}
```

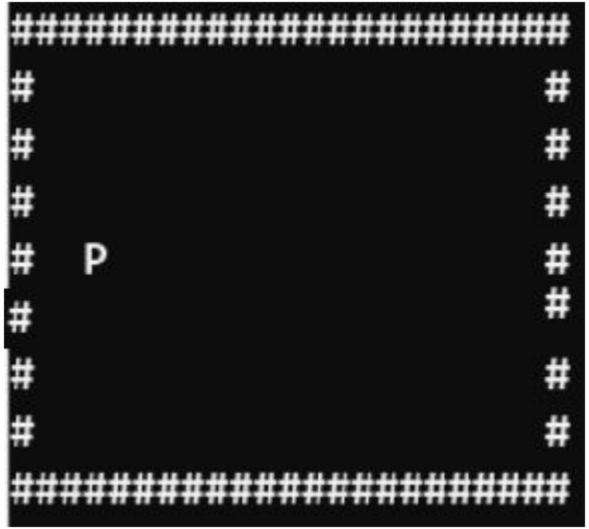


```
########################
#                      #
#                      #
#                      #
#  P                   #
#                      #
#                      #
#                      #
########################


G:\Semesters\Programming Fundamentals
```

Step 3: Move the player on Console

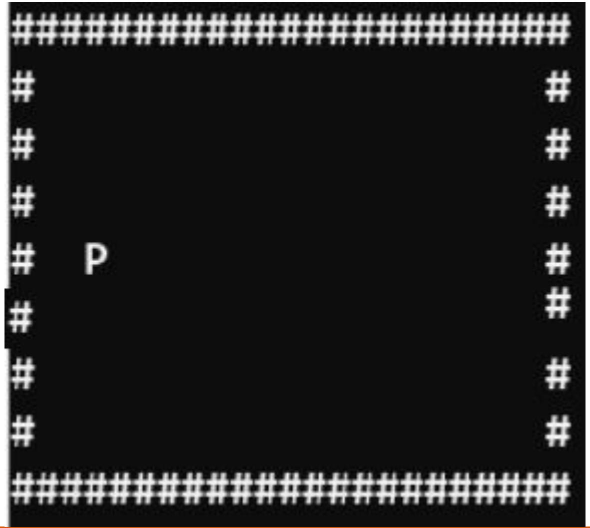
Now we have printed the Player on the specific location. If we want to move the player on the next location towards right then what will we have to do?



```
#######################
#                     #
#                     #
#                     #
#  P                  #
#                     #
#                     #
#                     #
#######################
```

Step 3: Move the player on Console

Now we have printed the Player on the specific location.
If we want to move the player on the next location towards right then what will we have to do?
We have to increment in the X coordinate.



```
########################
#                      #
#                      #
#                      #
#  P                   #
#                      #
#                      #
#                      #
########################
```

Step 3: Move the player on Console

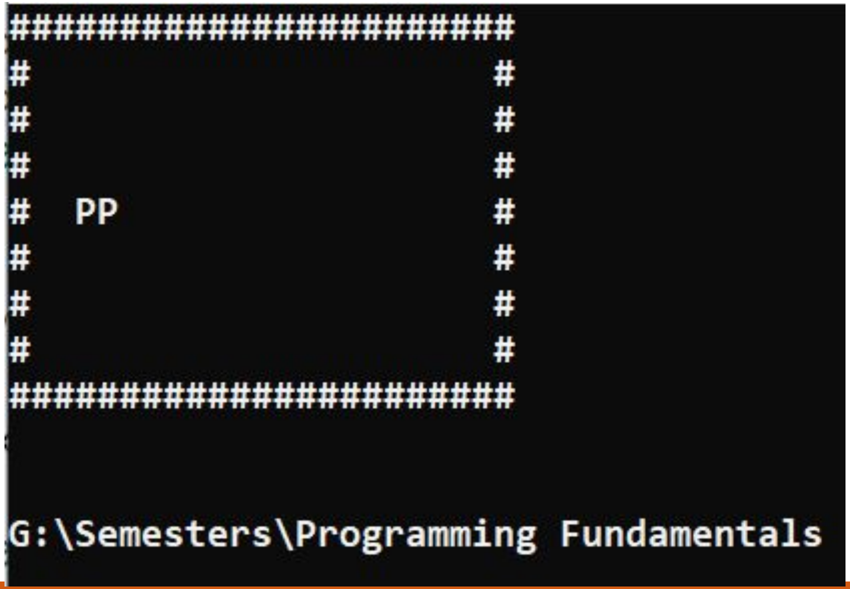
```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

Now, there are 2 players on the screen.



```
#######################
#                     #
#                     #
#                     #
#  PP                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

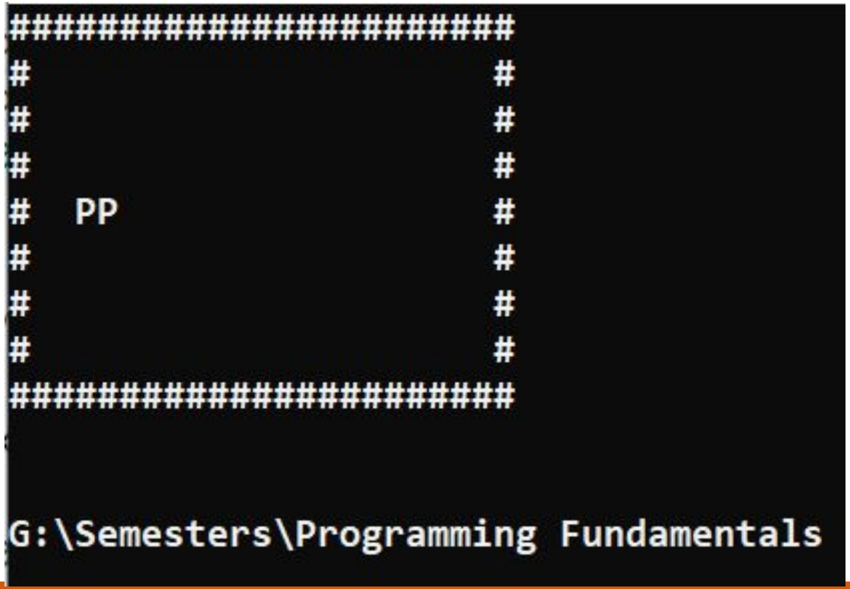
```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

Now, there are 2 players on the screen.
What we have to do to resolve it?



```
#######################
#                     #
#                     #
#                     #
#  PP                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

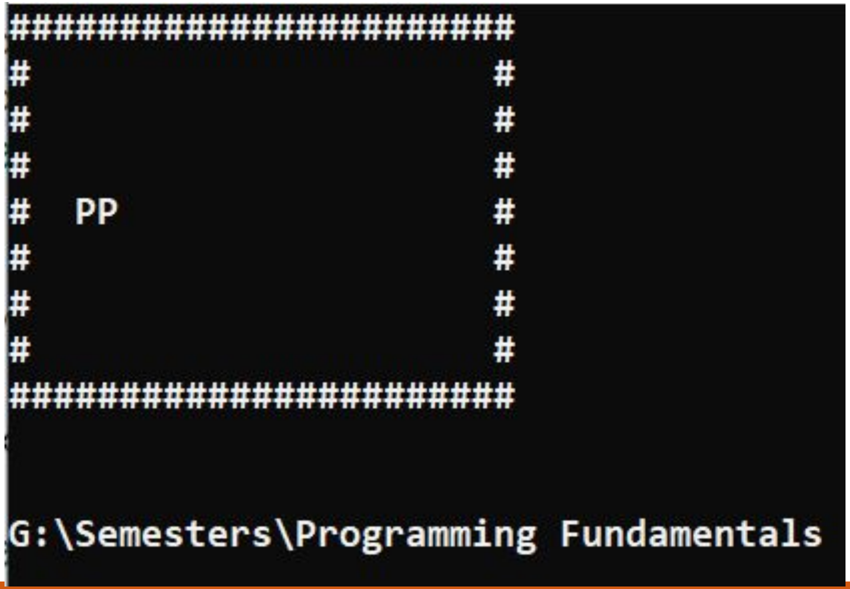
```cpp
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

We can remove the previous P by printing “ ” on the console on (3,4) location.



```
#######################
#                     #
#                     #
#                     #
#  PP                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);

}
```

Step 3: Move the player on Console

We can remove the previous P by printing “ ” on the console on (3,4) location.

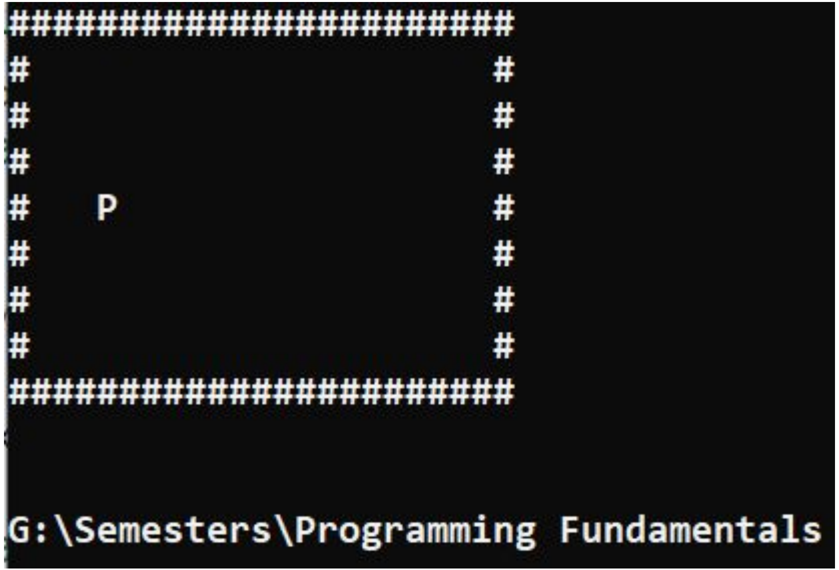
```cpp
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
     system("cls");
     printMaze();
     gotoxy(3, 4);
     cout << "P";
     gotoxy(3, 4);
     cout << " ";
     gotoxy(4, 4);
     cout << "P";
      gotoxy(0, 10);
}
```

Step 3: Move the player on Console

We can remove the previous P by printing " " on the console on (3,4) location.



```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

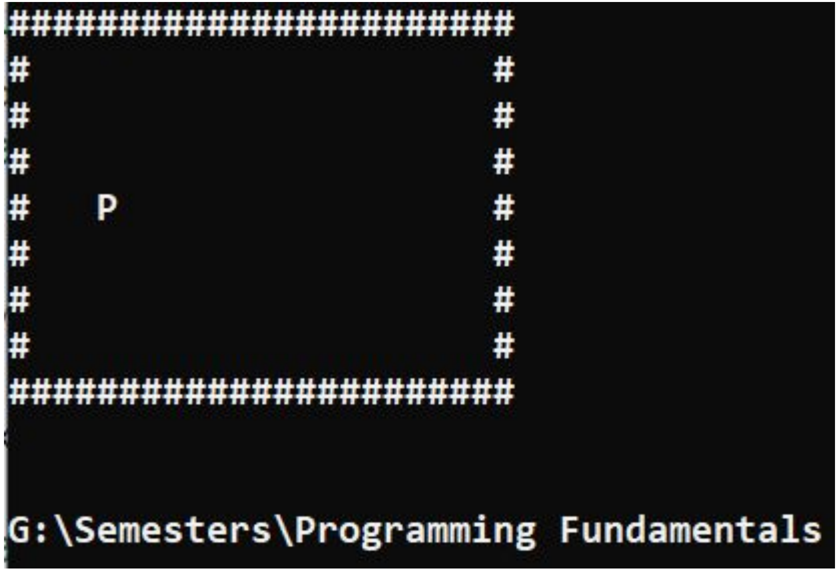
```cpp
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(3, 4);
    cout << " ";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

But it happened so fast that it was not giving the moving effect.



```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

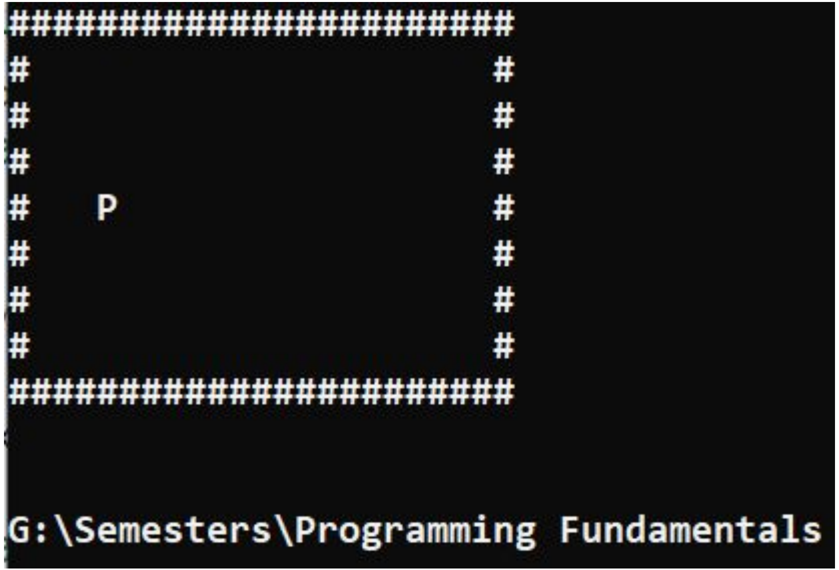
```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(3, 4);
    cout << " ";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

What can we do to delay the execution of some lines of code?



```
########################
#                      #
#                      #
#                      #
#   P                  #
#                      #
#                      #
#                      #
########################


G:\Semesters\Programming Fundamentals
```

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(3, 4);
    cout << " ";
    gotoxy(4, 4);
    cout << "P";
    gotoxy(0, 10);
}
```

Step 3: Move the player on Console

What can we do to delay the execution of some lines of code?

We can use sleep function

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    gotoxy(3, 4);
    cout << " ";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

What can we do to delay the execution of some lines of code?

We can use sleep function

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    Sleep(200);
    gotoxy(3, 4);
    cout << " ";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

Now, can we make a generic formula that will keep track of the coordinates instead of giving hard coded values to gotoxy function?

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    system("cls");
    printMaze();
    gotoxy(3, 4);
    cout << "P";
    Sleep(200);
    gotoxy(3, 4);
    cout << " ";
    gotoxy(4, 4);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

We can make 2 variables for X and Y.

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    gotoxy(x, y);
    cout << "P";
    Sleep(200);
    gotoxy(x, y);
    cout << " ";
    x = x + 1;
    gotoxy(x, y);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

We can make 2 variables for X and Y.
Now can you see some code repeating for moving the player on the console?

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    gotoxy(x, y);
    cout << "P";
    Sleep(200);
    gotoxy(x, y);
    cout << " ";
    x = x + 1;
    gotoxy(x, y);
    cout << "P";
     gotoxy(0, 10);
}
```

Step 3: Move the player on Console

We can make a move function to reuse it again and again.

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();

main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    gotoxy(x, y);
    cout << "P";
    Sleep(200);
    gotoxy(x, y);
    cout << " ";
    x = x + 1;
    gotoxy(x, y);
    cout << "P";
     gotoxy(0, 10);
}
```

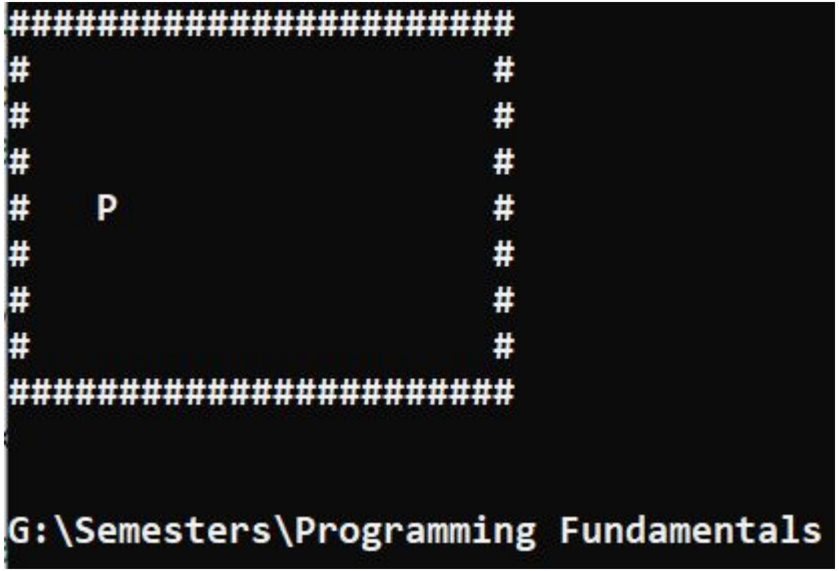
Step 3:

We can make a move function to reuse it again and again.

```
#include <iostream>
#include <windows.h>
using namespace std;
void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    playerMove(x,y);
    x = x+1;
    playerMove(x,y);
     gotoxy(0, 10);
}
void playerMove(int x, int y)
{
    gotoxy(x, y);
    cout << "P";
    Sleep(200);
    gotoxy(x, y);
    cout << " ";
}
```

Step 4: Keep Printing the player

But the program is terminated after printing the player 2 times.

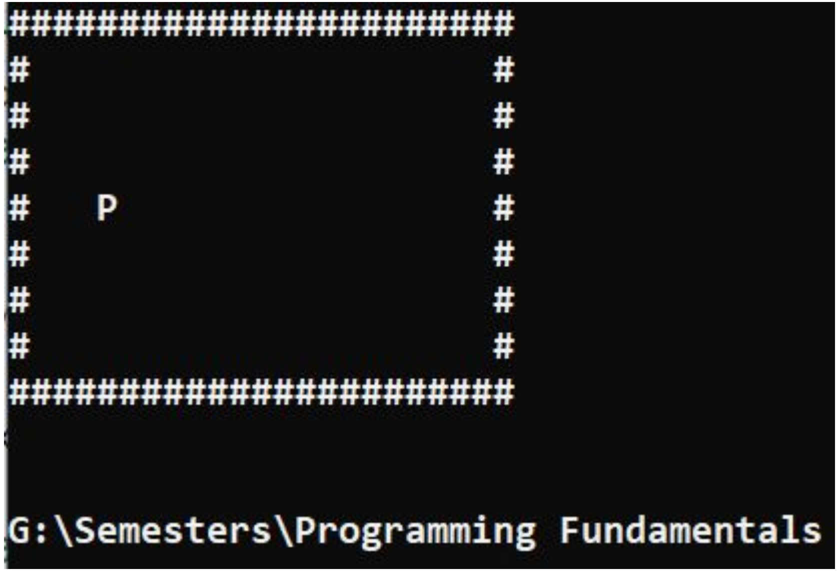


```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

Step 4: Keep Printing the player

If we want to keep the program running until closed forcefully, then what we have to do?

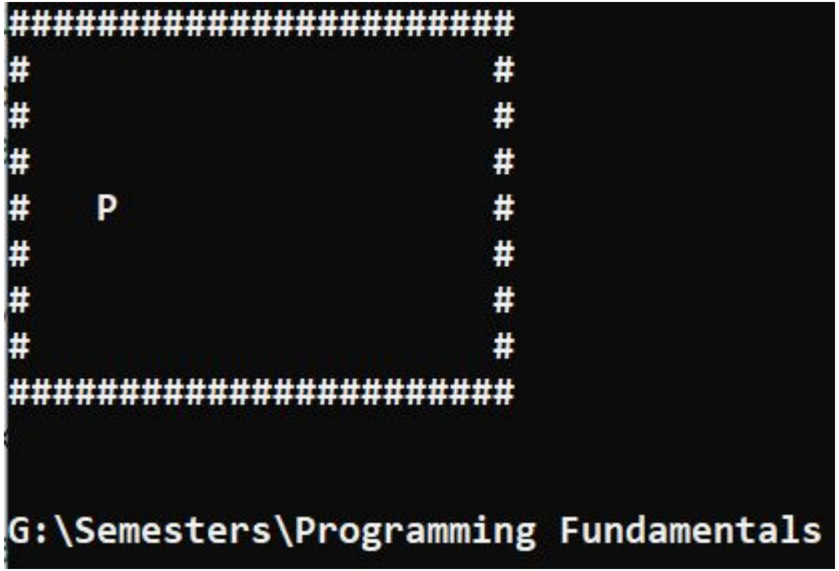


```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

Step 4: Keep Printing the player

We have to use the **while** loop.



```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################


G:\Semesters\Programming Fundamentals
```

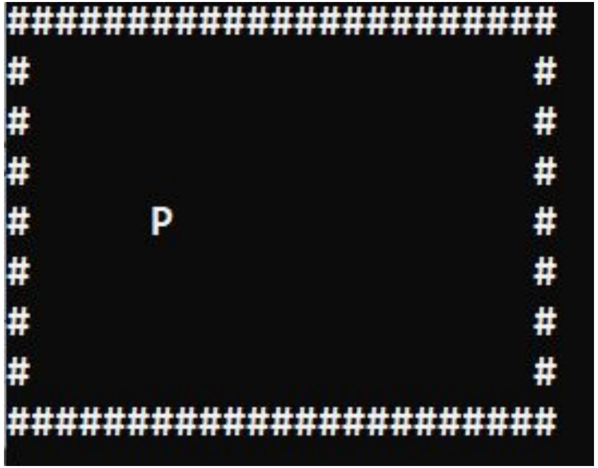
Step 4: Keep Printing the player

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    while (true)
    {
        system("cls");
        printMaze();
        playerMove(x, y);
        x = x + 1;
    }
    gotoxy(0, 10);
}
```

Step 4: Keep Printing the player

Now, it is taking some time to print the maze and the player. Why is it?



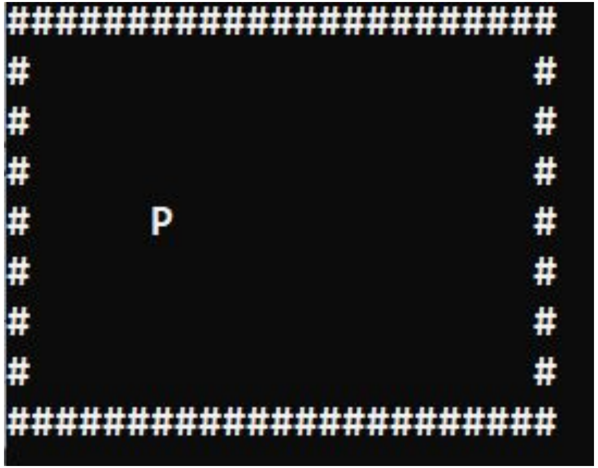
```
#######################
#                     #
#                     #
#                     #
#     P               #
#                     #
#                     #
#                     #
#######################
```

```cpp
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    while (true)
    {
        system("cls");
        printMaze();
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 4: Keep Printing the player

Now, it is taking some time to print the maze and the player. Why is it? How can we improve?



```
#######################
#                     #
#                     #
#                     #
#     P               #
#                     #
#                     #
#                     #
#######################
```

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    while (true)
    {
        system("cls");
        printMaze();
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 4: Keep Printing the player

We can print the maze only one time before the while loop.

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    while (true)
    {
        system("cls");
        printMaze();
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 4: Keep Printing the player

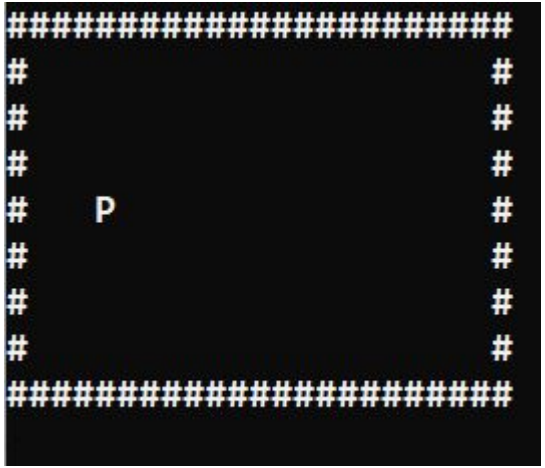
We can print the maze only one time before the while loop.

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    while (true)
    {
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 4: Keep Printing the player

We can print the maze only one time before the while loop.



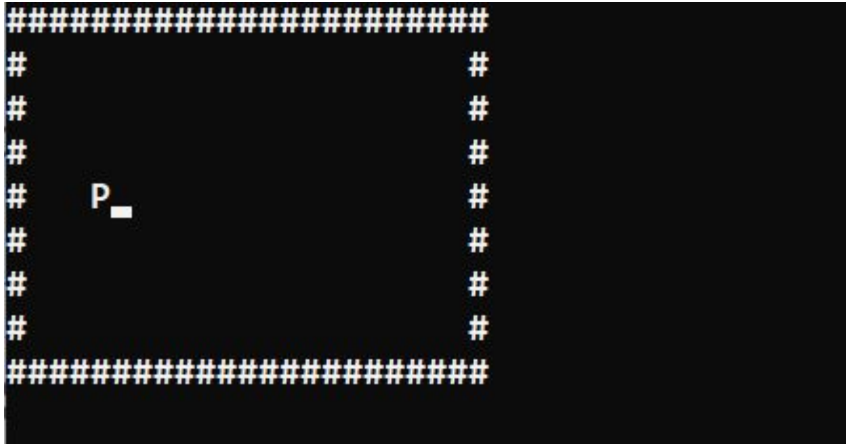
```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################
```

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    while (true)
    {
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 5: Player going outside maze

There is one issue with the code.



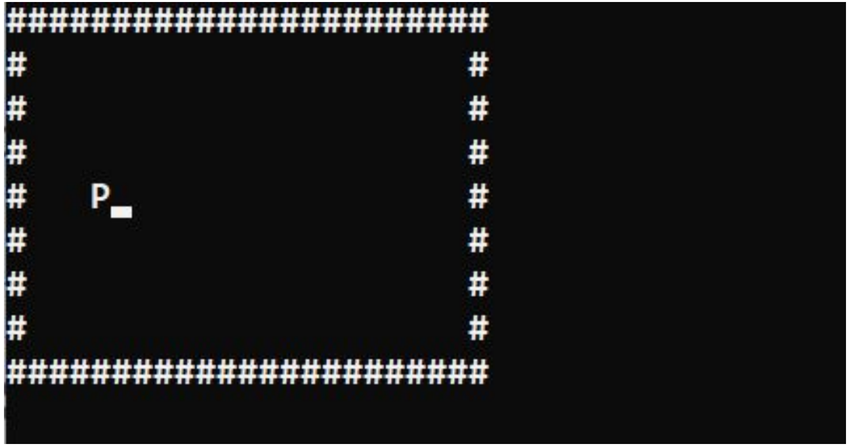
```
##########################
#                        #
#                        #
#                        #
#   P_                   #
#                        #
#                        #
#                        #
##########################
```

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    while (true)
    {
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 5: Player going outside maze

How to keep the player within the boundary of the maze?



```
#######################
#                     #
#                     #
#                     #
#   P                 #
#                     #
#                     #
#                     #
#######################
```

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    while (true)
    {
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 5: Player going outside maze

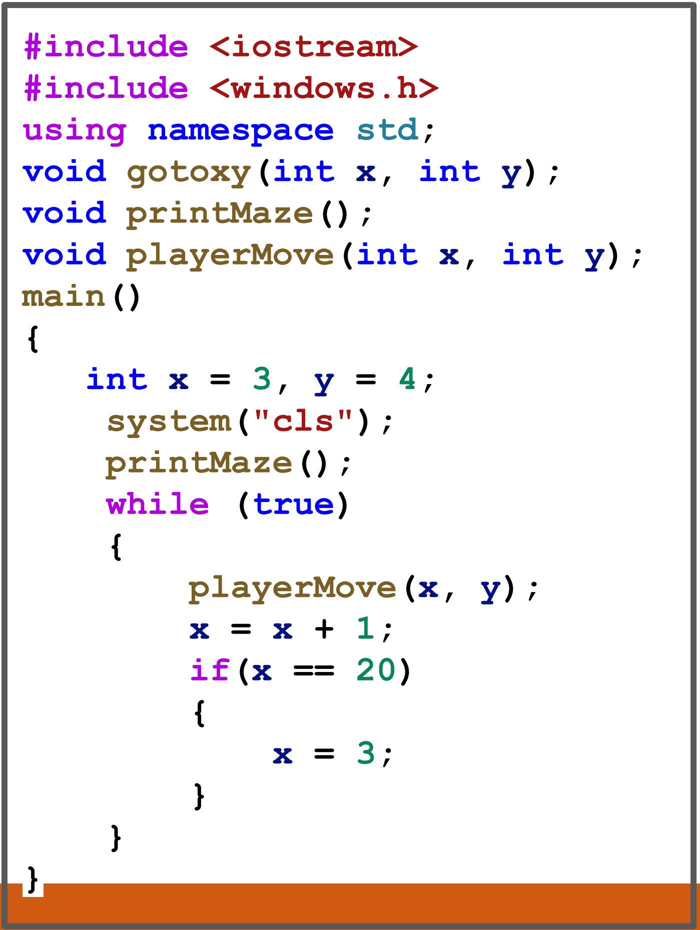
We can add a stopping condition, when the X coordinate of the player reaches some point we initialize it.

```
#include <iostream>
#include <windows.h>
using namespace std;

void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
    int x = 3, y = 4;
    system("cls");
    printMaze();
    while (true)
    {
        playerMove(x, y);
        x = x + 1;
    }
}
```

Step 5: Player going outside maze

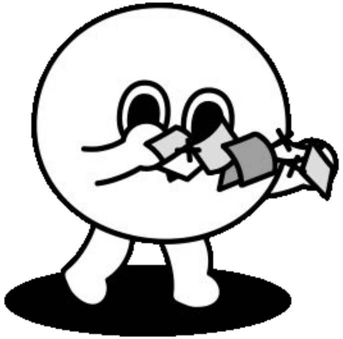
We can add a stopping condition, when the X coordinate of the player reaches some point we initialize it.

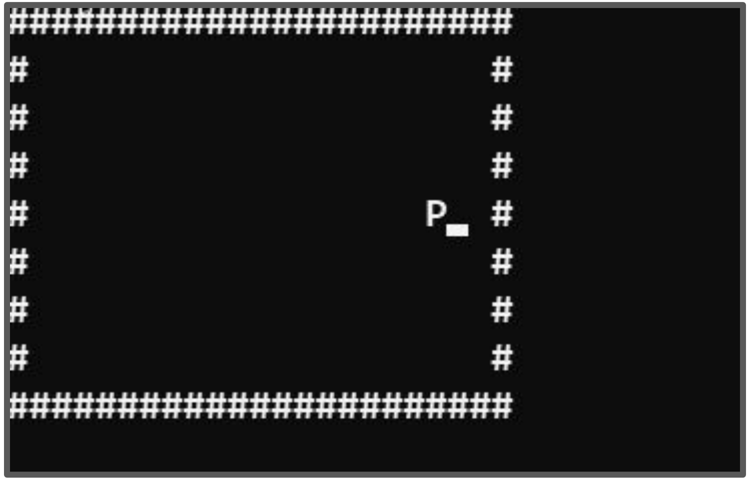


```
#include <iostream>
#include <windows.h>
using namespace std;
void gotoxy(int x, int y);
void printMaze();
void playerMove(int x, int y);
main()
{
   int x = 3, y = 4;
    system("cls");
    printMaze();
    while (true)
    {
        playerMove(x, y);
        x = x + 1;
        if(x == 20)
        {
            x = 3;
        }
    }
}
```

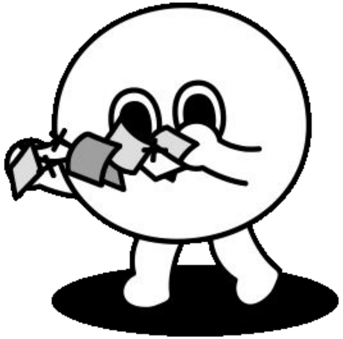
Goal: Move Pacman on the Console

Finally, the goal is achieved.





```
############################
#                          #
#                          #
#                          #
#                      P_  #
#                          #
#                          #
#                          #
############################
```



Learning Objective

Write a **C++ program** to display output on the console at a specific location using **gotoxy()** function.

Self Assessment

You have to write the Vision of your Business Application and Game that you want to develop in your 1st semester.