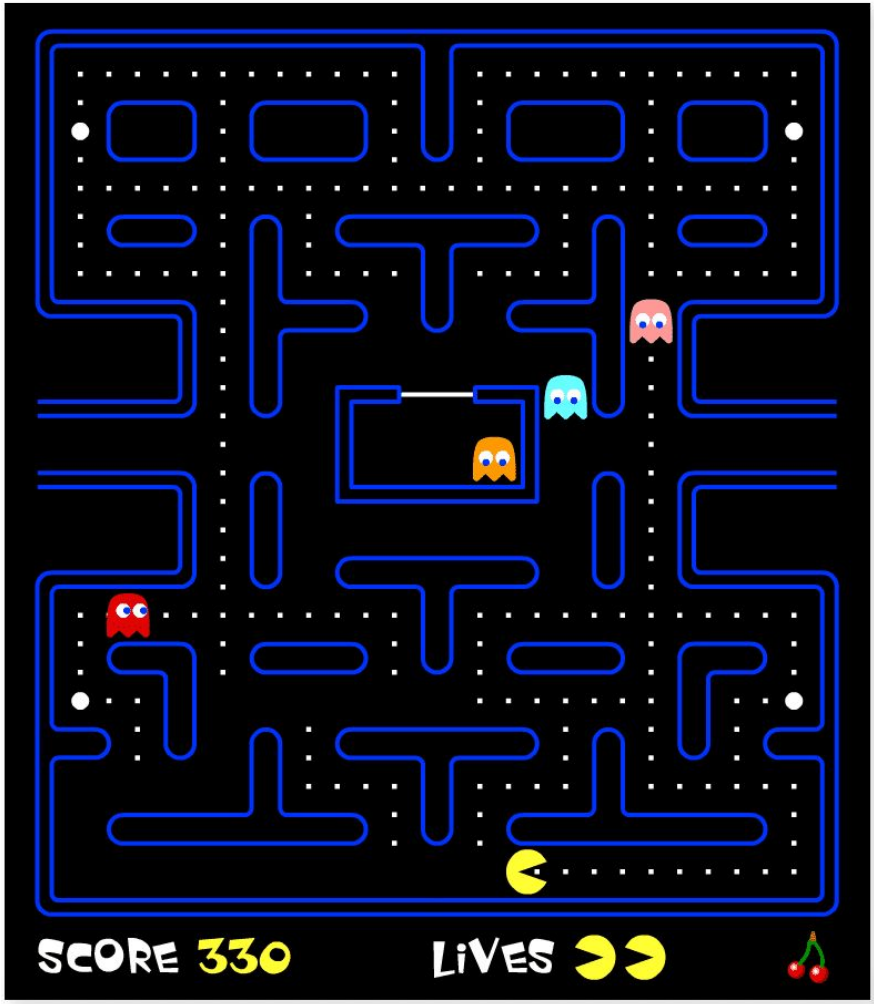
Game Development

اَللّٰهُمَّ اُرْزُقْنِى عِلْمًا نَافِعًا وَاسِعًا عَمِيْقًا

اَللّٰهُمَّ اُرْزُقْنِى رِزْقًا وَاسِعًا حَلَالًا طَيِّبًا
مُبَارَكًا مِنْ عِنْدِكَ

Pac-Man

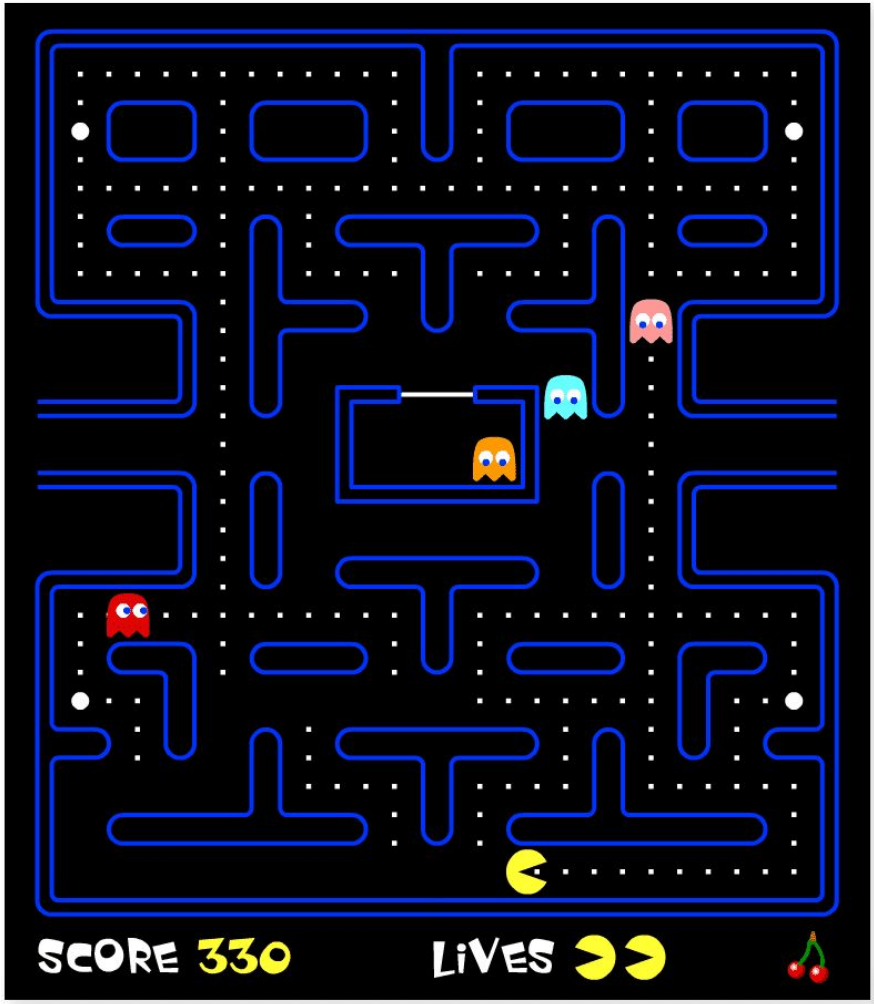
Pac-Man is a maze-based 2D game which was developed by Namco and first released in Japan on May 22, 1980.



Characters

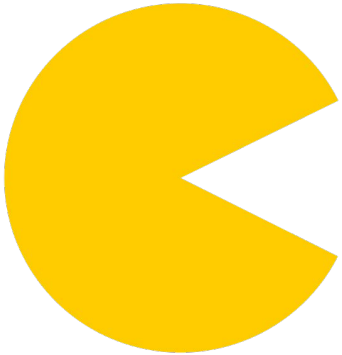
There are a total 5 characters in the Pac-Man Game.

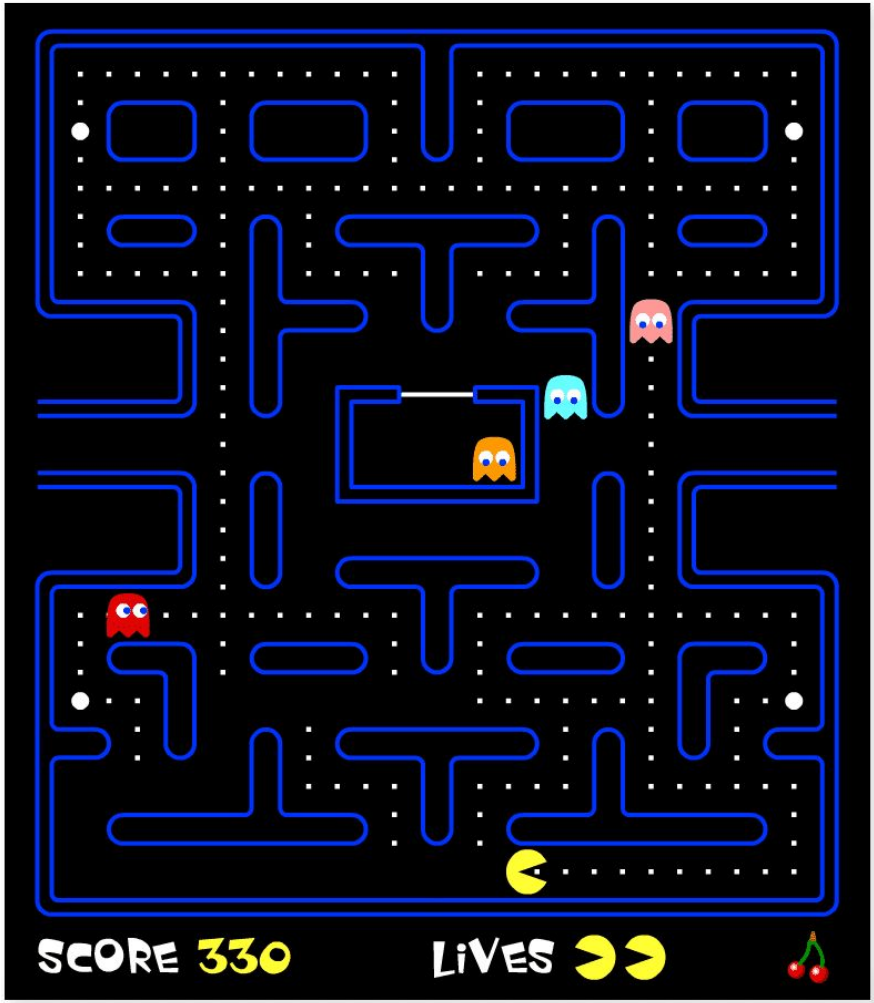
1 Pac-Man.
4 Ghosts.



Characters: Pac-Man

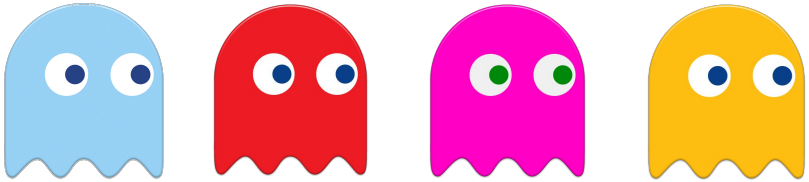
A yellow, circular character named Pac-Man is controlled by the player with the help of arrow keys.

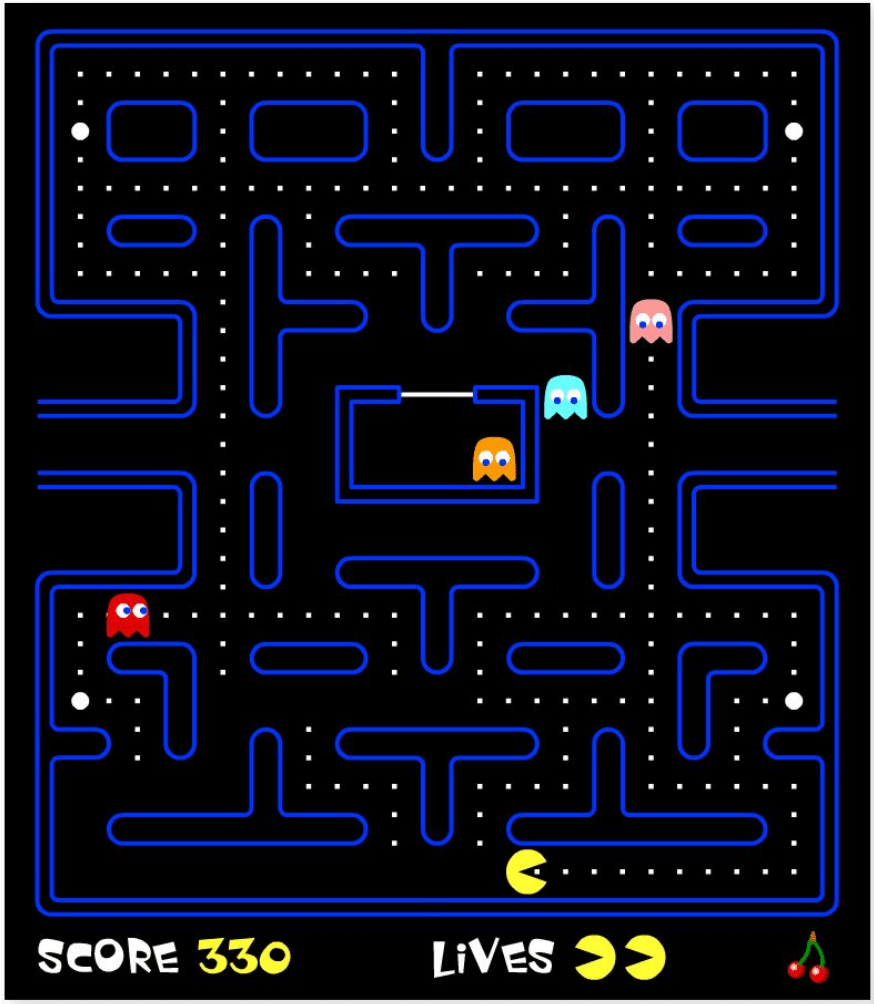




Characters: Ghosts

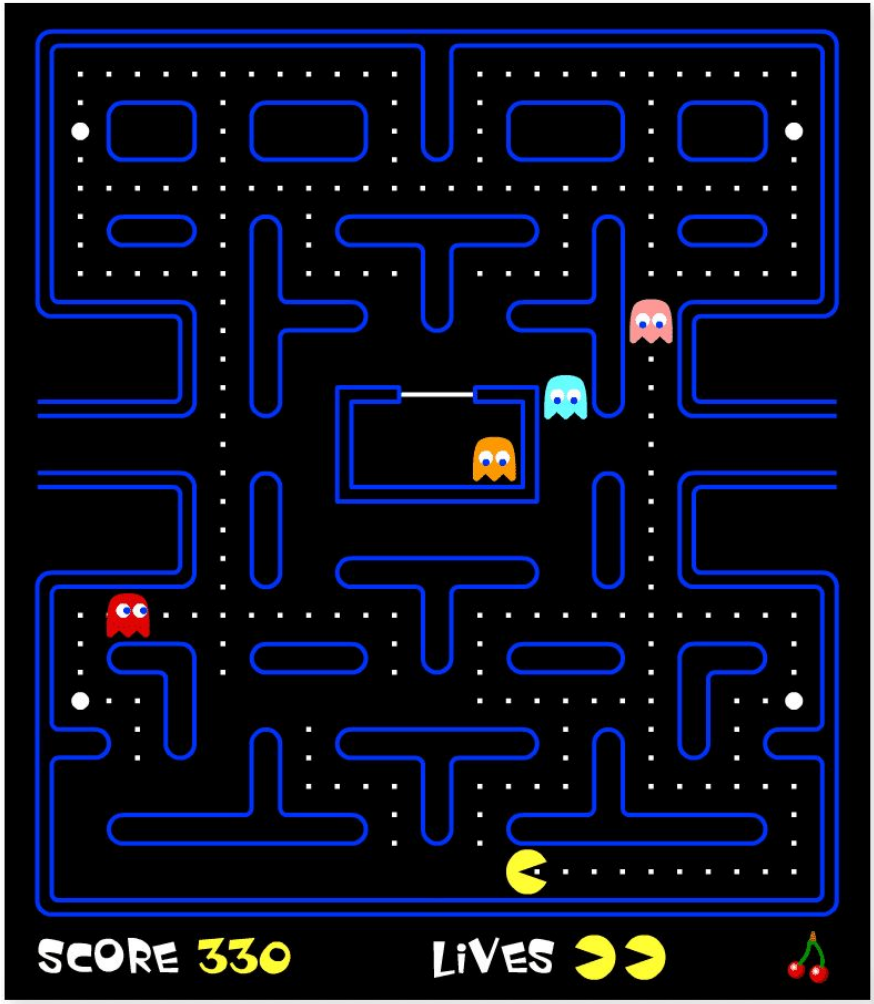
Each of the four ghosts are controlled by the computer.





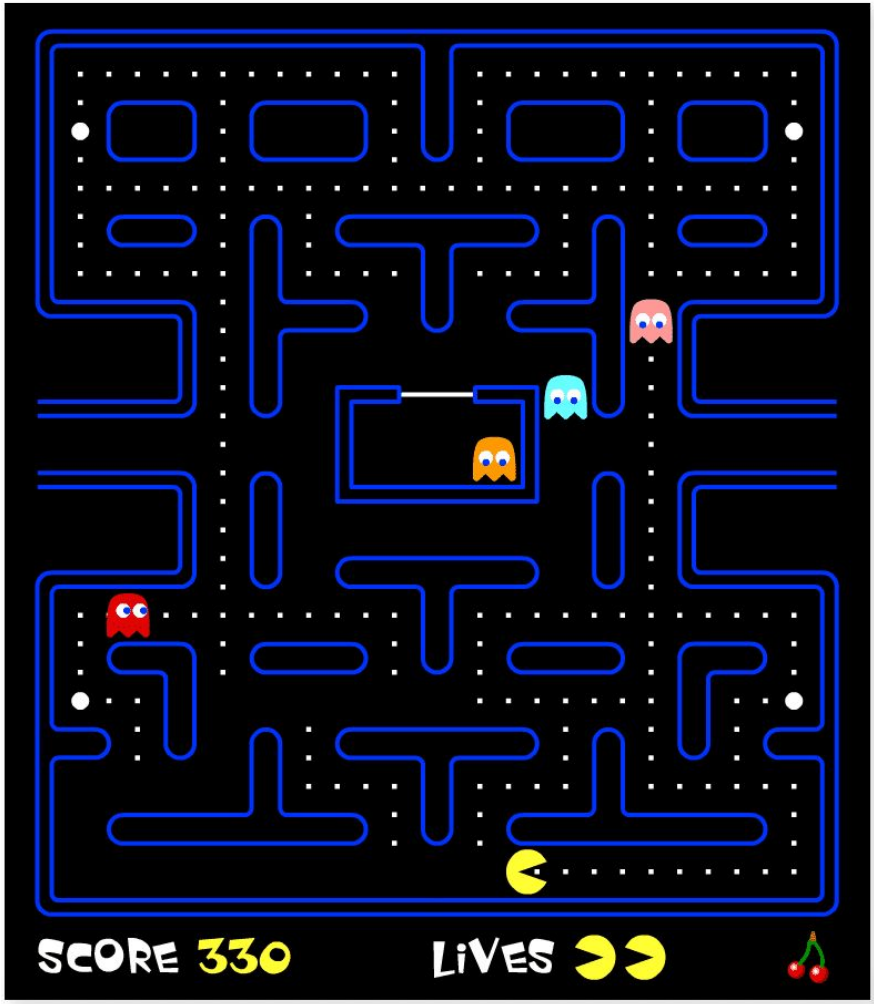
Objects: Food Pallets

Small white dots are called "Food Pallets" whereas the large flashing white dots are called "Power Pallets" or "Energizers".



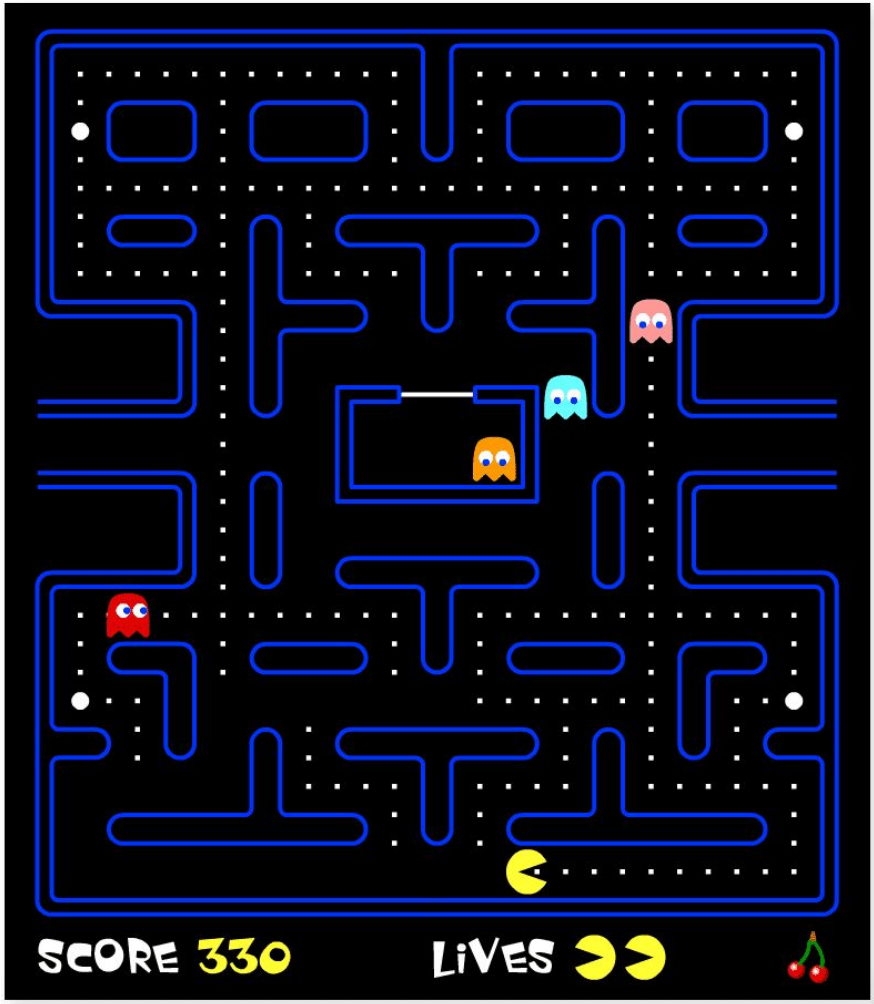
Objects: Walls

Blue outline represents the walls of the maze.



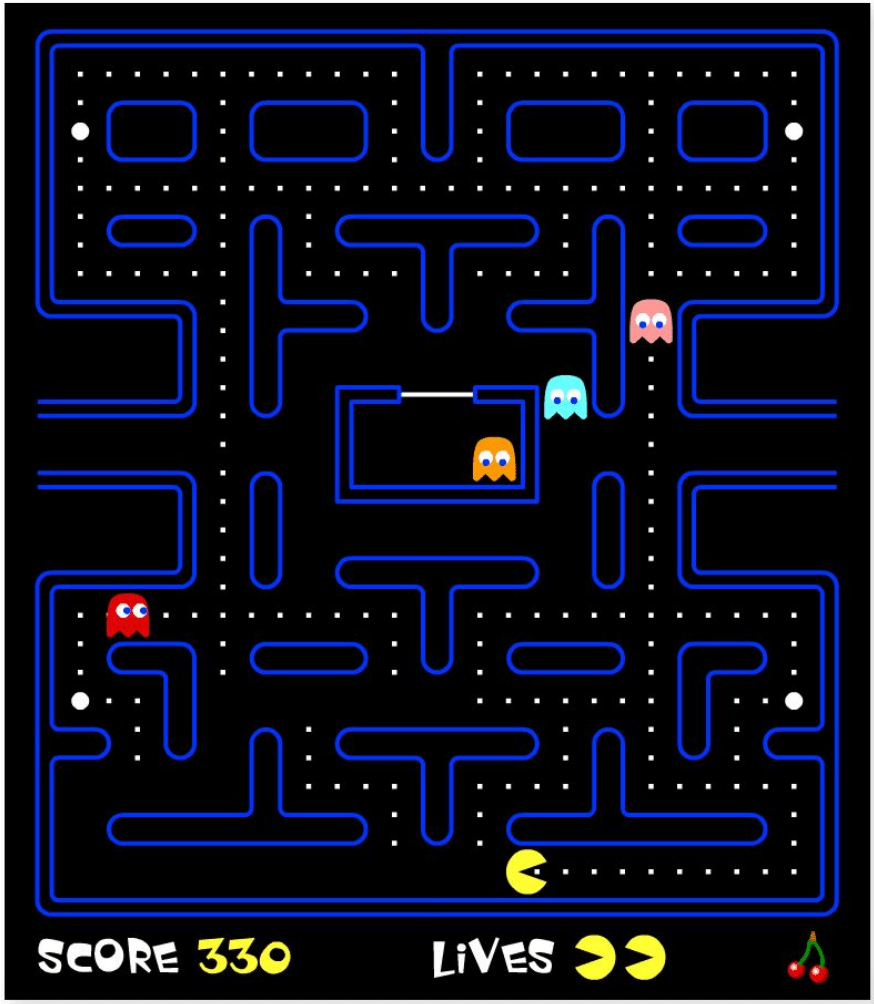
Rules & Interactions

Pac-Man can eat food pallets that have been put across the maze.
Pac-Man loses a life if he collides with any of the ghosts.
If Pac-man eats Power Pallets then the ghosts will turn blue and then Pac-Man can touch the ghosts as well.



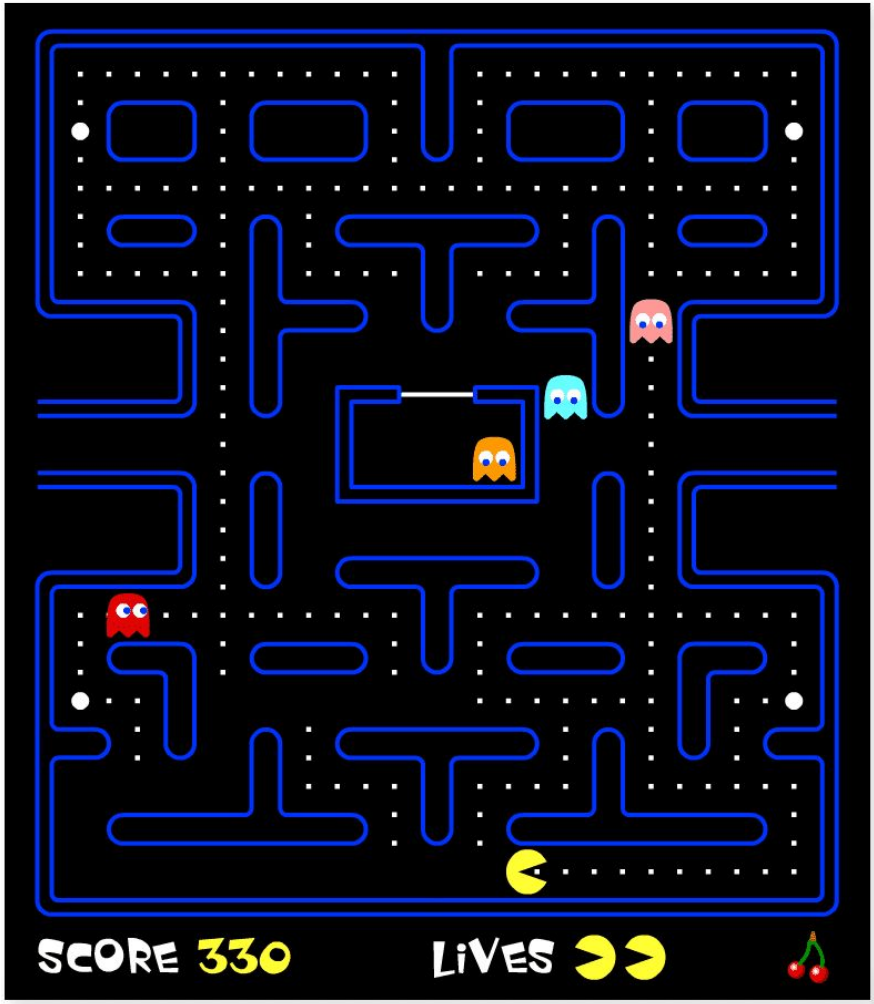
Rules & Interactions

Score increases when the Pac-Man eats food pallets.



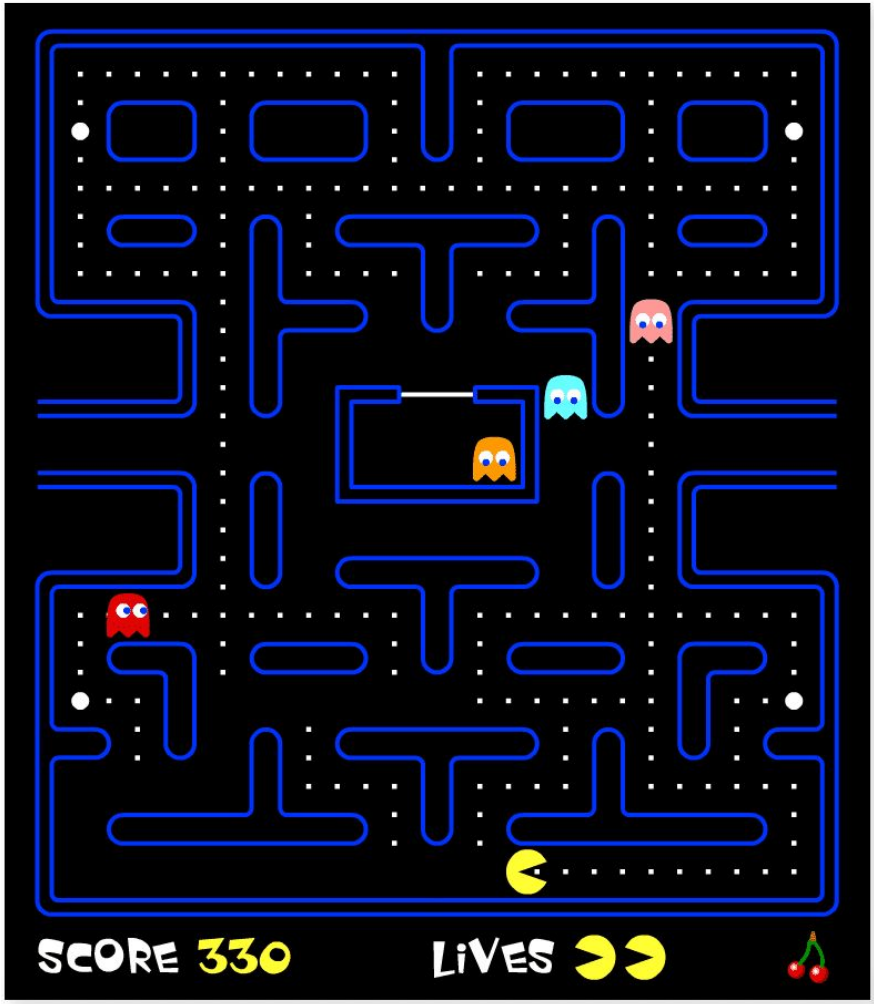
Goal

The goal of the game is to eat all of the food pallets that have been put across the maze while avoiding the Ghosts.



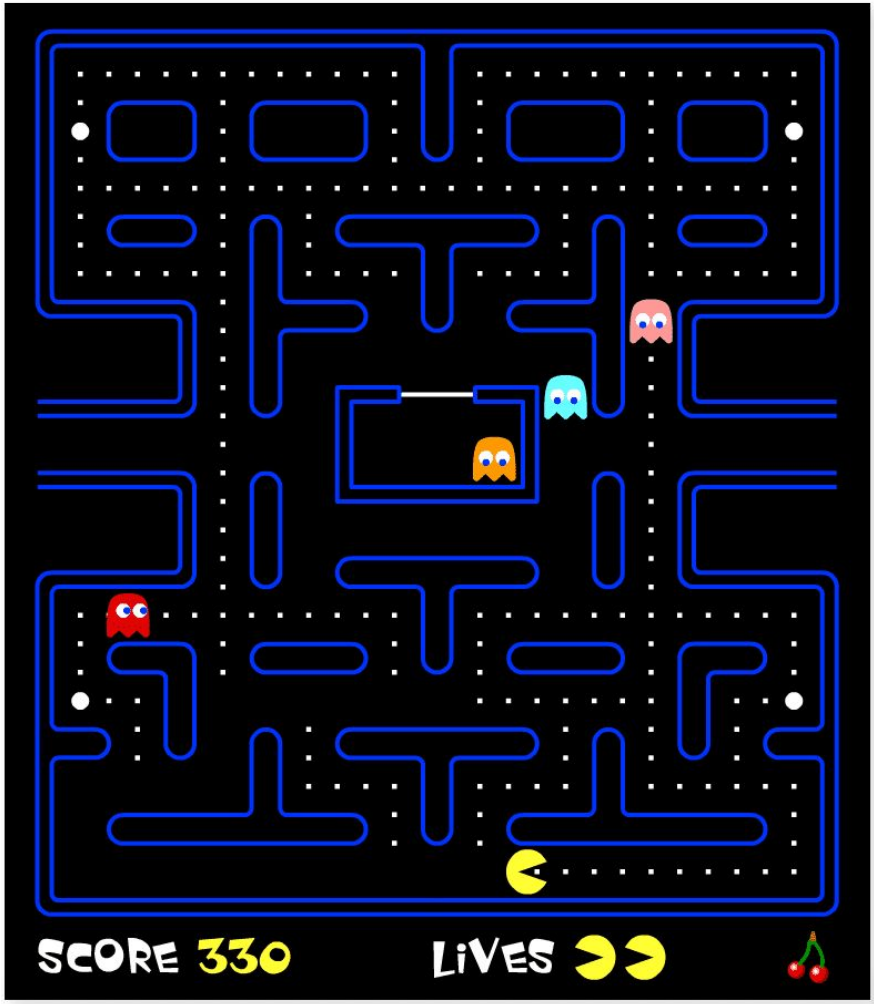
GUI Based Game

This is a GUI-based Game.



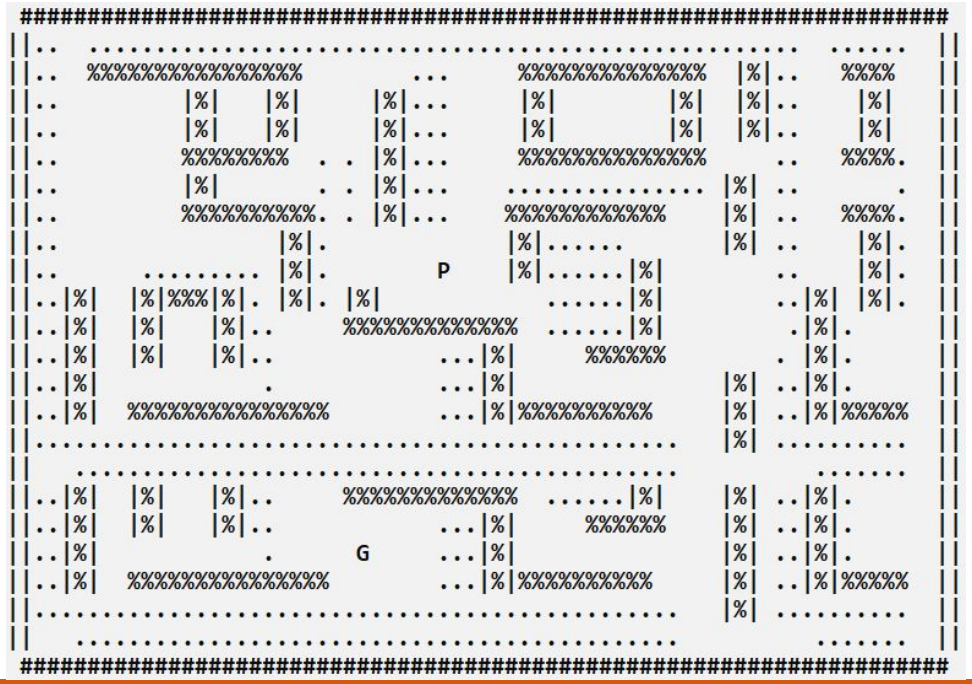
GUI Based Game

This is a GUI-based Game. We will develop console based game for now.



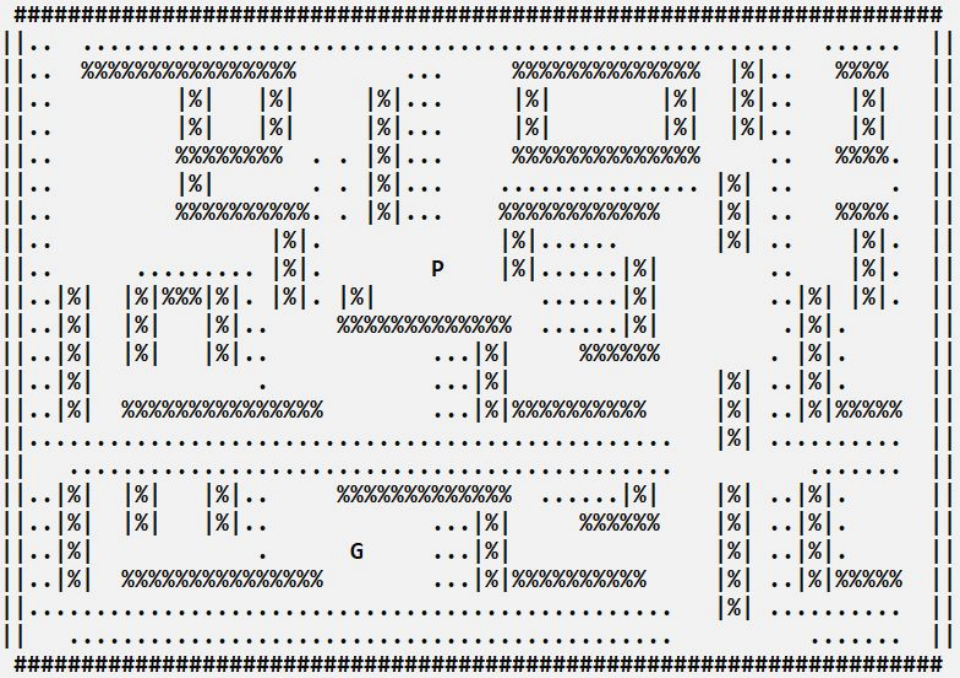
Console Based Game

The Logic behind both GUI and CLI game is the same which is the most important.



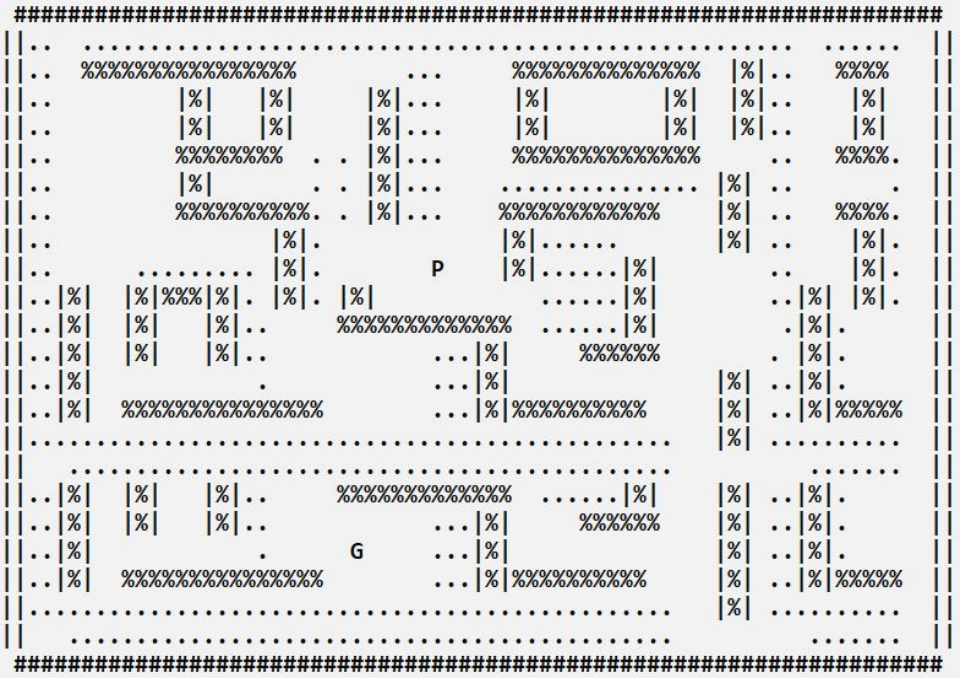
Console Based Game

Pac-Man is represented by P, ghost by G, Walls with |, # and %



Console Based Game

We have already printed this maze on the Console.

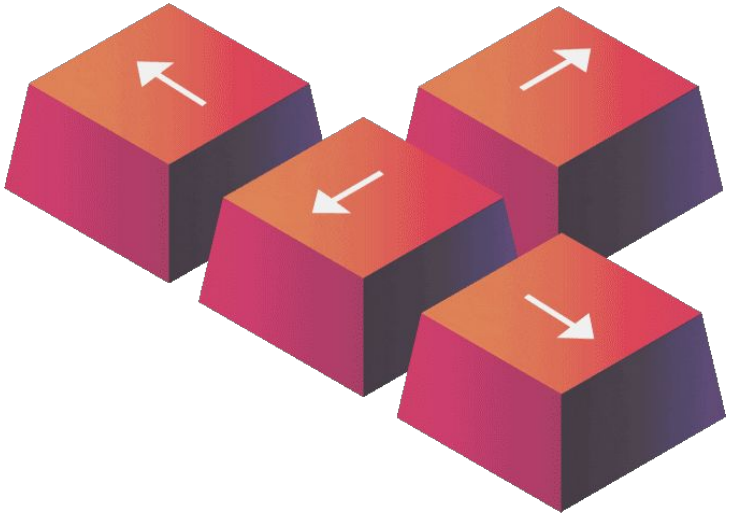


Moving Pac-Man

We have moved the Pac-man in the maze horizontally as well as vertically.

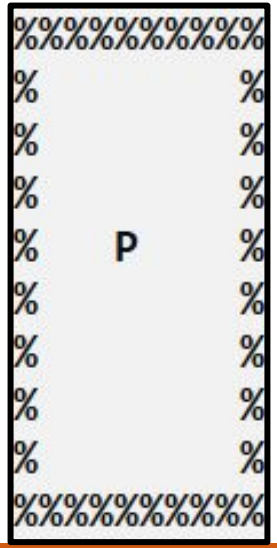
Moving Pac-Man using Arrow Keys

Most important thing is to make Pac-Man move with the help of arrow keys.



Moving Pac-Man using Arrow Keys

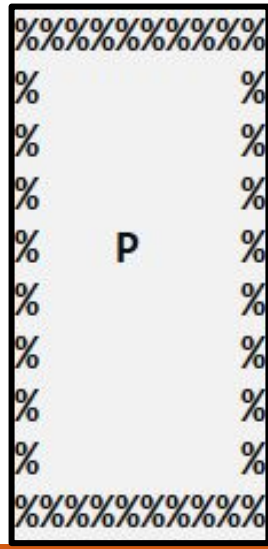
Most important thing is to make Pac-Man move with the help of arrow keys.
For Simplicity, let's make a 10x10 Pac-Man game with Pac-Man present at 4x4 Location.



Moving Pac-Man

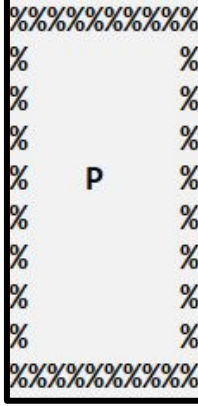
Most important thing is to make Pac-Man move with the help of arrow keys.
For Simplicity, let's make a 10x10 Pac-Man game with Pac-Man present at 4x4 Location.
Player can press:

- Left arrow key (Move Left)
- Right arrow key (Move Right)
- Up arrow key (Move Up)
- Down arrow key (Move Down)

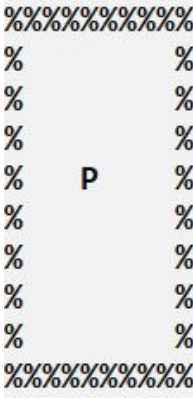


Pac-Man:Representation on Console

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%				P					%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%



Pac-Man: Move Left

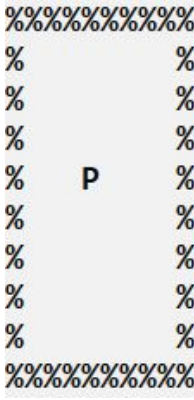


	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%			P ←						%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

y Same

x - 1

Pac-Man: Move Right



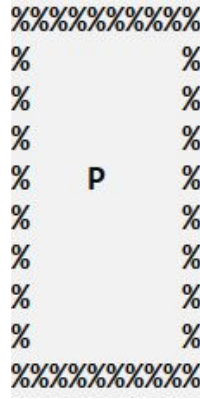
	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%					P				%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

y Same

x + 1

Pac-Man: Move UP

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%				P					%
4	%									%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

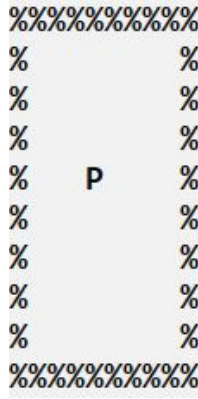


y - 1

x Same

Pac-Man: Move Down

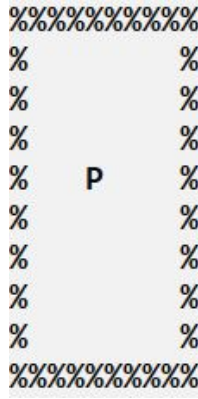
	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%									%
5	%				P					%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%



y + 1

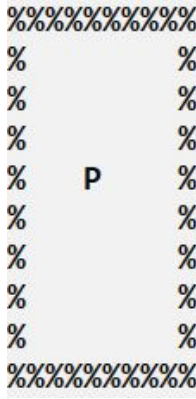
x Same

Pac-Man: Movement



Important thing to note here is Pac-Man is removed from the previous location when it is moved to the next location.

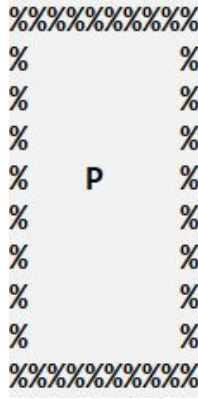
Pac-Man: Movement



We can make the following general formulas

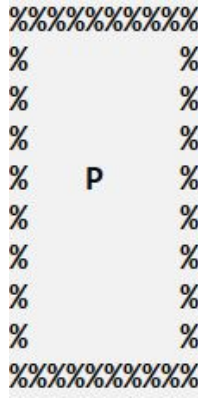
Keys	Movement
Up	y decrements by 1, x remains same
Down	y increments by 1, x remains same
Left	y remains same, x decrements by 1
Right	y remains same, x increments by 1

Pac-Man: Detect Arrow Key



Before changing the location of the Pac-Man we have to detect which arrow key is pressed

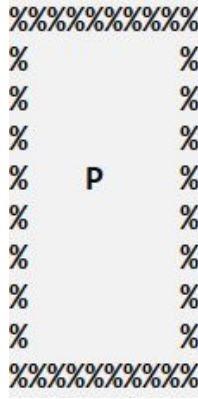
Pac-Man: Detect Arrow Key



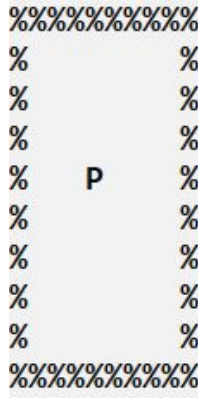
C++ provides us with a function named GetAsyncKeyState().
GetAsyncKeyState stands for Get Asynchronous Key State. This function gives information about the key, whether the key was pressed or not at the time when the function was called.

Pac-Man: Detect Arrow Key

But we have to only detect arrow keys.



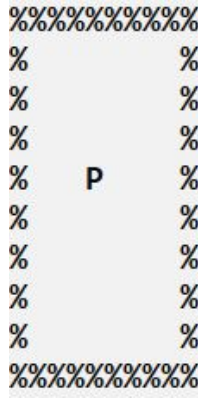
Pac-Man: Virtual Key Codes



C++ also provides Virtual-key code constants that are used to find the state of the pressed keys.

Codes	Meaning
VK_LEFT	Left arrow key
VK_RIGHT	Right arrow key
VK_UP	Up Arrow key
VK_DOWN	Down arrow key

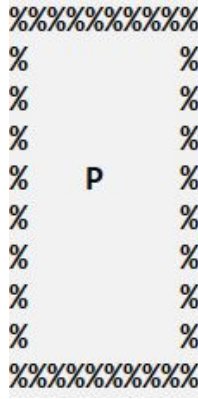
Pac-Man: Detect Up arrow Key



Code to detect if the up arrow key is pressed is given by:

```
1  if (GetAsyncKeyState(VK_UP))
2  {
3      // Move the Pac Man Up
4  }
```

Pac-Man: Detect Up arrow Key

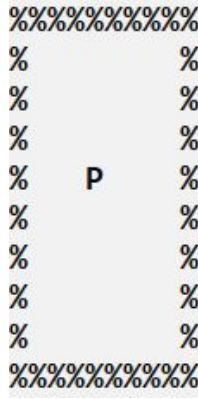


Code to detect if the up arrow key is pressed is given by:

```
1  if (GetAsyncKeyState(VK_UP))
2  {
3      // Move the Pac Man Up
4  }
```

GetAsyncKeyState(VK_UP) function returns 0 if the key is not pressed and a non zero value if the key is currently pressed.

Pac-Man: windows.h

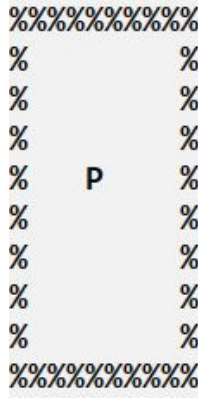


The definition of GetAsyncKeyState() function is given in the windows.h header file.

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

```
#include <iostream>
#include <windows.h>
using namespace std;
          // Function Prototype
void printMaze();
void gotoxy(int x, int y);
main(){
    int pacmanX = 4; // X Coordinate of Pacman
    int pacmanY = 4; // Y Coordinate of Pacman
    bool gameRunning = true;
    system("cls");
    printMaze();
    gotoxy(pacmanX, pacmanY);
    cout << "P";
    while (gameRunning){
        if (GetAsyncKeyState(VK_LEFT)){
        }
        if (GetAsyncKeyState(VK_RIGHT)){
        }
        if (GetAsyncKeyState(VK_UP)){
        }
        if (GetAsyncKeyState(VK_DOWN)){
        }
        if (GetAsyncKeyState(VK_ESCAPE)){
            gameRunning = false;
        }
        Sleep(200);
    }
}
```

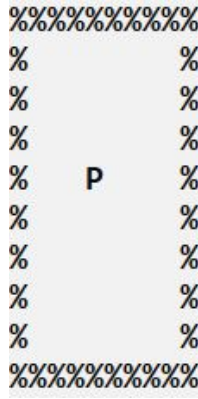
Pac-Man: Activity



Now, your task is to implement the following functionalities.

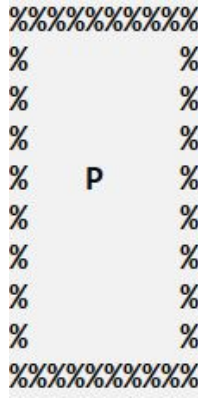
1. void printMaze();
2. if (GetAsyncKeyState(VK_LEFT))
3. if (GetAsyncKeyState(VK_RIGHT))
4. if (GetAsyncKeyState(VK_UP))
5. if (GetAsyncKeyState(VK_DOWN)

Pac-Man: PrintMaze()



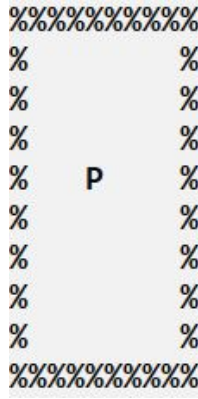

```
void printMaze()
{
    cout << "%%%%%%%%%%" << endl;
    cout << "%          %" << endl;
    cout << "%          %" << endl;
    cout << "%          %" << endl;
    cout << "%          %" << endl;
    cout << "%          %" << endl;
    cout << "%          %" << endl;
    cout << "%          %" 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

Pac-Man: movePacmanLeft



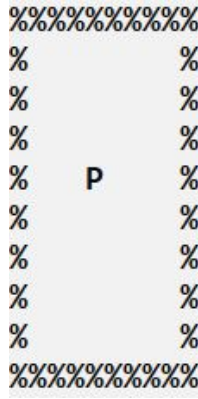

```
if (GetAsyncKeyState(VK_LEFT))
{
     gotoxy(pacmanX, pacmanY);
     cout << " ";
     pacmanX = pacmanX - 1;
     gotoxy(pacmanX, pacmanY);
     cout << "P";
}
```

Pac-Man: movePacmanRight



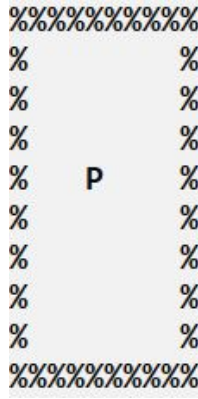

```
if (GetAsyncKeyState(VK_RIGHT))
{
     gotoxy(pacmanX, pacmanY);
     cout << " ";
     pacmanX = pacmanX + 1;
     gotoxy(pacmanX, pacmanY);
     cout << "P";
}
```

Pac-Man: movePacmanUp




```
if (GetAsyncKeyState(VK_UP))
{
     gotoxy(pacmanX, pacmanY);
     cout << " ";
     pacmanY = pacmanY - 1;
     gotoxy(pacmanX, pacmanY);
     cout << "P";
}
```

Pac-Man: movePacmanDown




```
if (GetAsyncKeyState(VK_DOWN))
{
     gotoxy(pacmanX, pacmanY);
     cout << " ";
     pacmanY = pacmanY + 1;
     gotoxy(pacmanX, pacmanY);
     cout << "P";
}
```

Pac-Man: Activity

Do you see any code repeating in the move functions?

```
if (GetAsyncKeyState(VK_LEFT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_RIGHT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_UP))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_DOWN))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

Pac-Man: Code Repetition

To clear the Pacman from the previous location.

```
if (GetAsyncKeyState(VK_LEFT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_RIGHT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_UP))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_DOWN))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

Pac-Man: Code Repetition

To show the Pacman on the updated location.

```
if (GetAsyncKeyState(VK_LEFT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_RIGHT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_UP))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_DOWN))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

Pac-Man: Solution?

What is the solution to avoid the code repetition?

```
if (GetAsyncKeyState(VK_LEFT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_RIGHT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_UP))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_DOWN))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

Pac-Man: Solution

We can make separate functions.

```
if (GetAsyncKeyState(VK_LEFT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_RIGHT))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanX = pacmanX + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_UP))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY - 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

```
if (GetAsyncKeyState(VK_DOWN))
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
    pacmanY = pacmanY + 1;
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

Pac-Man: Solution

We can make separate functions to erase and print Pacman.

```
void erasePacman()
{
    gotoxy(pacmanX, pacmanY);
    cout << " ";
}
```

```
void printPacman()
{
    gotoxy(pacmanX, pacmanY);
    cout << "P";
}
```

Pac-Man: Solution

Updated Move functionalities.

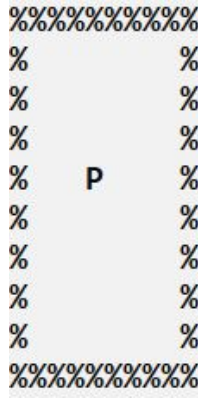
```
if (GetAsyncKeyState(VK_LEFT))
{
    erasePacman();
    pacmanX = pacmanX - 1;
    printPacman();
}
```

```
if (GetAsyncKeyState(VK_RIGHT))
{
    erasePacman();
    pacmanX = pacmanX + 1;
    printPacman();
}
```

```
if (GetAsyncKeyState(VK_UP))
{
    erasePacman();
    pacmanY = pacmanY - 1;
    printPacman();
}
```

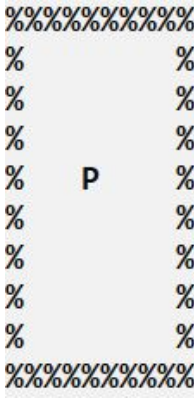
```
if (GetAsyncKeyState(VK_DOWN))
{
    erasePacman();
    pacmanY = pacmanY + 1;
    printPacman();
}
```

Pac-Man: Activity



Do you see any other problem in this code?

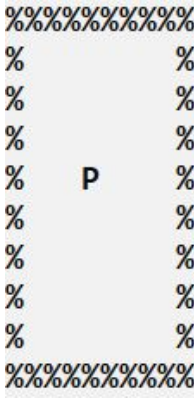
Pac-Man: Collision with Wall



What Happens when Pac-Man reaches any wall?

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%	P								%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

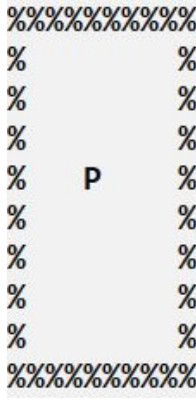
Pac-Man: Collision with Wall



% will be replaced with P.

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%	P								%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

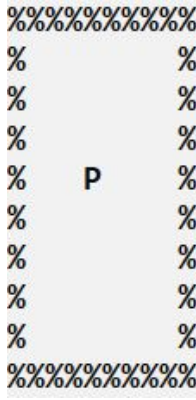
Pac-Man: Collision with Wall



So, what can be the solution of this?

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%	P								%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

Pac-Man: Collision with Wall



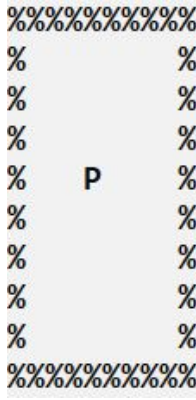
Before moving to the next place, we should know what is present on that location.

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%	P								%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

Pac-Man: Collision with Wall

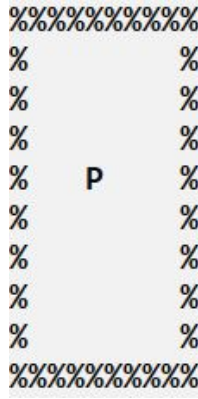
Then we should only move if the next place contains empty space.

	0	1	2	3	4	5	6	7	8	9
0	%	%	%	%	%	%	%	%	%	%
1	%									%
2	%									%
3	%									%
4	%	P								%
5	%									%
6	%									%
7	%									%
8	%									%
9	%	%	%	%	%	%	%	%	%	%

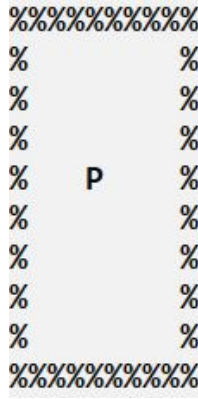


Pac-Man: Collision with Wall

In C++, we have a function that will read the character present on console on specific coordinates.



Pac-Man: Collision with Wall



In C++, we have a function that will read the character present on console on specific coordinates.

```
char getCharAtxy(short int x, short int y)
{
    CHAR_INFO ci;
    COORD xy = {0, 0};
    SMALL_RECT rect = {x, y, x, y};
    COORD coordBufSize;
    coordBufSize.X = 1;
    coordBufSize.Y = 1;
    return ReadConsoleOutput(GetStdHandle(STD_OUTPUT_HANDLE), &ci, coordBufSize, xy, &rect) ? ci.Char.AsciiChar
: ' ';
}
```

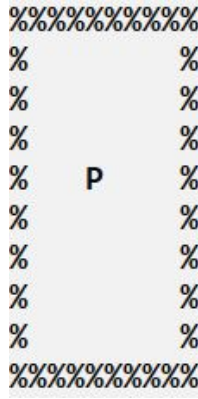
Pac-Man: Collision with Wall

```
%%%%%%%%%%
%        %
%        %
%        %
%   P    %
%        %
%        %
%        %
%        %
%%%%%%%%%%
```

Now, our updated Move functionalities will become

```
if (GetAsyncKeyState(VK_RIGHT))
{
    char nextLocation = getCharAtxy(pacmanX + 1, pacmanY);
    if (nextLocation == ' ')
    {
        erasePacman();
        pacmanX = pacmanX + 1;
        printPacman();
    }
}
```

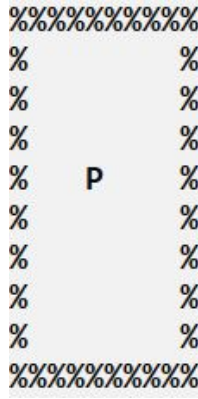
Pac-Man: Collision with Wall



Now, our updated Move functionalities will become

```
if (GetAsyncKeyState(VK_LEFT))
{
    char nextLocation = getCharAtxy(pacmanX - 1, pacmanY);
    if (nextLocation == ' ')
    {
        erasePacman();
        pacmanX = pacmanX - 1;
        printPacman();
    }
}
```

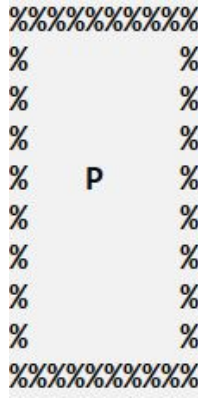
Pac-Man: Collision with Wall



Now, our updated Move functionalities will become

```
if (GetAsyncKeyState(VK_UP))
{
    char nextLocation = getCharAtxy(pacmanX, pacmanY - 1);
    if (nextLocation == ' ')
    {
        erasePacman();
        pacmanY = pacmanY - 1;
        printPacman();
    }
}
```

Pac-Man: Collision with Wall



Now, our updated Move functionalities will become

```
if (GetAsyncKeyState(VK_DOWN))
{
    char nextLocation = getCharAtxy(pacmanX, pacmanY + 1);
    if (nextLocation == ' ')
    {
        erasePacman();
        pacmanY = pacmanY + 1;
        printPacman();
    }
}
```

Learning Objective

Write a C++ program to move a game object on the console using arrow keys and detect collision.

Conclusion

- GetAsyncKeyState stands for Get Asynchronous Key State. This function gives information whether the key was pressed or not at the time when the function was called.
- Virtual key codes for the arrow keys and their meanings are given below.

Codes	Meaning
VK_LEFT	Left arrow key
VK_RIGHT	Right arrow key
VK_UP	Up Arrow key
VK_DOWN	Down arrow key

Conclusion

- Syntax to use GetAsyncKeyState() function is as follows:

```
#include <windows.h>
main()
{
     if (GetAsyncKeyState(VK_Code))
     {
          // Do something
     }
}
```

Conclusion

- Function to read a character from the console is as follows:

```
char getCharAtxy(short int x, short int y)
{
    CHAR_INFO ci;
    COORD xy = {0, 0};
    SMALL_RECT rect = {x, y, x, y};
    COORD coordBufSize;
    coordBufSize.X = 1;
    coordBufSize.Y = 1;
    return ReadConsoleOutput(GetStdHandle(STD_OUTPUT_HANDLE), &ci, coordBufSize, xy, &rect) ? ci.Char.AsciiChar
: ' ';
}
```

Self Assessment: (Video Profile Activity)

1. Now your task is to make the Pac-Man for a larger grid world. Include the functionality, if the Pac-Man strucks the wall, it does not change its position.

2. Also, Add 2 Ghosts in the Grid and move one of them vertically and other horizontally.

3. Also, add the functionality if Pac-Man collides with Ghost game should be over, and if it collides with food pallets then its score increases.