Programming Fundamental

Programming Day - Week 11

Introduction

Welcome to your favorite day of the week which is programming day . In this lab manual, we shall work together to learn and implement new programming concepts.

Let's do some coding.

Introduction

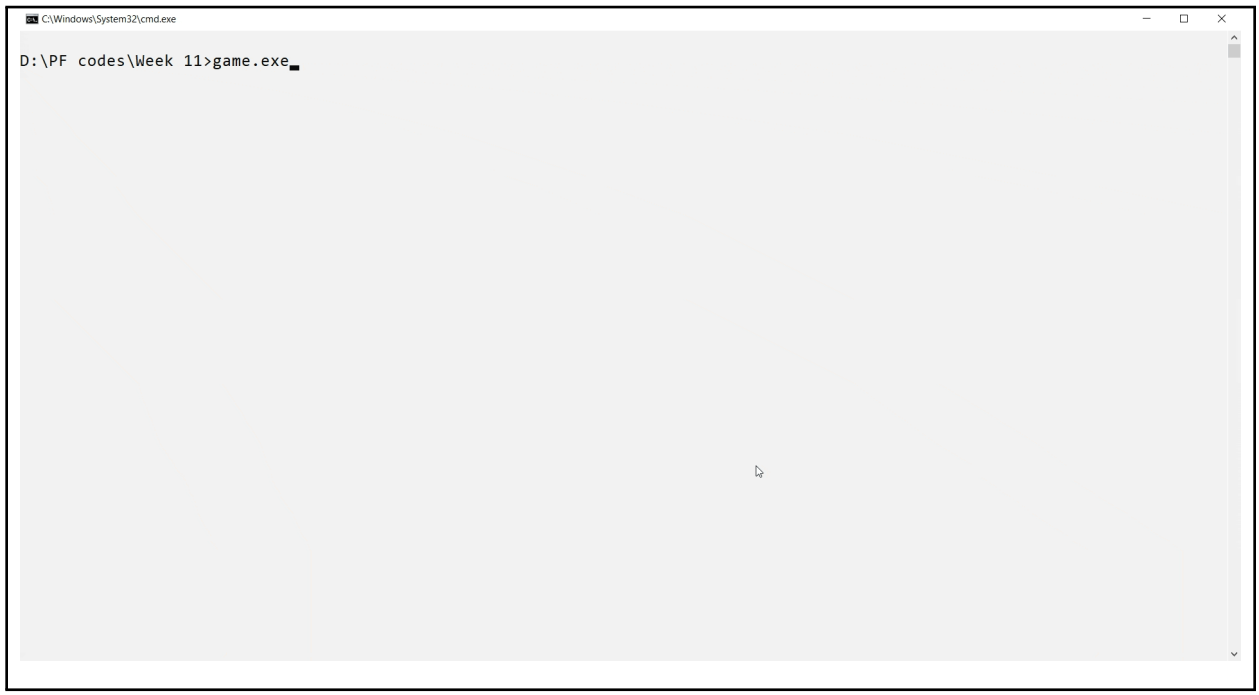
By this week, you have learned how to write a program that contains functions, loops, arrays and conditional structures. In this class, we will learn permanently store the data into the computer and how to decompose difficult problems into small sets of easy problems and then solve them easily.

Consider that we want to develop a game that the characters as Tanks where we have a player tank and three enemy tanks and the enemy dies after collision with the fire generated by the player and vise versa while the score of the game increases.
Lets execute our idea of the game one step at a time.

Consider this **mini game** with the above mentioned features for better understanding.

Programming Fundamental

Programming Day - Week 11



Lets code it out !

Step 01: Print Maze and Characters

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

```
    // Player Character
char box = 219;
char tank1[6] = {box, box, box, '-', '-', '>'};
char tank2[6] = {'0', ' ', '0', ' ', ' ', ' '};

    // Enemy Character
char enemy1[6] = {' ', ' ', ' ', '-', '-', '-'};
char enemy2[6] = {'<', '=', '=', '(', '-', ')'};
char enemy3[6] = {' ', ' ', ' ', '\\', '@', '/'};
char enemy4[6] = {' ', ' ', ' ', '*', '*', '*'};

    // Player Coordinates
int tankX = 5;
int tankY = 5;

    // Enemy Coordinates
int enemyX = 30;
int enemyY = 10;
void printTank()
{
    gotoxy(tankX, tankY);
    for (int index = 0; index < 6; index++)
    {
        cout << tank1[index];
    }
    gotoxy(tankX, tankY + 1);
    for (int index = 0; index < 6; index++)
    {
        cout << tank2[index];
    }
}
```

```
void printEnemy()
{
    gotoxy(enemyX, enemyY);
    for (int index = 0; index < 6; index++)
    {
        cout << enemy1[index];
    }
    gotoxy(enemyX, enemyY + 1);
    for (int index = 0; index < 6; index++)
    {
        cout << enemy2[index];
    }
    gotoxy(enemyX, enemyY + 2);
    for (int index = 0; index < 6; index++)
    {
        cout << enemy3[index];
    }
    gotoxy(enemyX, enemyY + 3);
    for (int index = 0; index < 6; index++)
    {
        cout << enemy4[index];
    }
}
```

Step 02: Character Movement

- **Player Movement**

```
void moveTankLeft()
{
    char next = getCharAtxy(tankX - 1, tankY);
    if (next == ' ')
    {
        eraseTank();
        tankX = tankX - 1;
        printTank();
    }
}
void moveTankRight()
{
    char next = getCharAtxy(tankX + 6, tankY);
    if (next == ' ')
    {
        eraseTank();
        tankX = tankX + 1;
        printTank();
    }
}
void moveTankUp()
{
    char next = getCharAtxy(tankX, tankY - 1);
    if (next == ' ')
    {
        eraseTank();
        tankY = tankY - 1;
        printTank();
    }
}
void moveTankDown()
{
    char next = getCharAtxy(tankX, tankY + 2);
    if (next == ' ')
    {
        eraseTank();
        tankY = tankY + 1;
        printTank();
    }
}
```

- **Enemy Movement**

```
void moveEnemy()
{
    if (enemyDirection == "Up")
    {
        char next = getCharAtxy(enemyX, enemyY - 1);
        if (next == ' ')
        {
            eraseEnemy();
            enemyY--;
            printEnemy();
        }
        if (next == '#')
        {
            enemyDirection = "Down";
        }
    }
    if (enemyDirection == "Down")
    {
        char next = getCharAtxy(enemyX, enemyY + 4);
        if (next == ' ')
        {
            eraseEnemy();
            enemyY++;
            printEnemy();
        }
        if (next == '#')
        {
            enemyDirection = "Up";
        }
    }
}
```

- **Supporting Functions**

```
void eraseEnemy()
{
    gotoxy(enemyX, enemyY);
    for (int index = 0; index < 6; index++)
    {
        cout << " ";
    }
    gotoxy(enemyX, enemyY + 1);
    for (int index = 0; index < 6; index++)
    {
        cout << " ";
    }
    gotoxy(enemyX, enemyY + 2);
    for (int index = 0; index < 6; index++)
    {
        cout << " ";
    }
    gotoxy(enemyX, enemyY + 3);
    for (int index = 0; index < 6; index++)
    {
        cout << " ";
    }
}
```

```
void eraseTank()
{
    gotoxy(tankX, tankY);
    for (int index = 0; index < 6; index++)
    {
        cout << " ";
    }
    gotoxy(tankX, tankY + 1);
    for (int index = 0; index < 6; index++)
    {
        cout << " ";
    }
}
```

Step 03: Firing/Shooting

- **Global Arrays and Variable**

```
    // Player Bullets
int bulletX[100];
int bulletY[100];
bool isBulletActive[100];
int bulletCount = 0;
```

- **Generate Bullet**

```
void generateBullet()
{
    bulletX[bulletCount] = tankX + 7;
    bulletY[bulletCount] = tankY;
    isBulletActive[bulletCount] = true;
    gotoxy(tankX + 7, tankY);
    cout << ".";
    bulletCount++;
}
```

- **Move Bullet**

```
void moveBullet()
{
    for (int x = 0; x < bulletCount; x++)
    {
        if (isBulletActive[x] == true)
        {
            char next = getCharAtxy(bulletX[x] + 1, bulletY[x]);
            if (next != ' ')
            {
                eraseBullet(bulletX[x], bulletY[x]);
                makeBulletInactive(x);
            }
            else
            {
                eraseBullet(bulletX[x], bulletY[x]);
                bulletX[x] = bulletX[x] + 1;
                printBullet(bulletX[x], bulletY[x]);
            }
        }
    }
}
```

- **Supporting Functions**

```
void printBullet(int x, int y)
{
    gotoxy(x, y);
    cout << ".";
}
void eraseBullet(int x, int y)
{
    gotoxy(x, y);
    cout << " ";
}
void makeBulletInactive(int index)
{
    isBulletActive[index] = false;
}
```

Step 04: Collision Detection

- **Collision With Enemy**

```
void bulletCollisionWithEnemy()
{
    for (int x = 0; x < bulletCount; x++)
    {
        if (isBulletActive[x] == true)
        {
            if (bulletX[x] + 1 == enemyX && (bulletY[x] == enemyY || bulletY[x] == enemyY + 2 || bulletY[x] == enemyY + 3))
            {
                addScore();
            }
            if (enemyX - 1 == bulletX[x] && enemyY + 1 == bulletY[x])
            {
                addScore();
            }
        }
    }
}
```

- **Reward/Score**

```
void addScore()
{
    score++;
}
void printScore()
{
    gotoxy(45, 8);
    cout << "Score: " << score;
}
```

- **Supporting Functions**

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

- **Main Function**

Programming Fundamental

Programming Day - Week 11

```
main()
{
    system("cls");
    printMaze();
    printTank();
    printEnemy();
    while (true)
    {
        printScore();
        if (GetAsyncKeyState(VK_LEFT))
        {
            moveTankLeft();
        }
        if (GetAsyncKeyState(VK_RIGHT))
        {
            moveTankRight();
        }
        if (GetAsyncKeyState(VK_UP))
        {
            moveTankUp();
        }
        if (GetAsyncKeyState(VK_DOWN))
        {
            moveTankDown();
        }
        if (GetAsyncKeyState(VK_SPACE))
        {
            generateBullet();
        }
        if (timer == 3)
        {
            moveEnemy();
            timer = 0;
        }
        moveBullet();
        bulletCollisionWithEnemy();
        timer++;
        Sleep(90);
    }
}
```

Good Luck and Best Wishes !!

Happy Coding ahead :)

Muhammad Irzam & Maida Shahid, Department of Computer Science, UET Lahore