**Object Oriented Programming**

**Assessment**

Name\_\_khadija imran\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Registration No: 2022Cs171\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Guideline:** You are only allowed to make changes in CollisionDetector.cs, form\_load (function), GameGhostHorizontal and GameGhostChaser.cs class.

**Past the code that you inserted in this document under every question**

**Q No 1: Increase score by one point if pacman collide with the pallet.**

Past yourif (collider.isPacManCollideWithPallet(potentialNewCell))

{

game.addScorePoints(1);

} code here

**Q No 2: Decrease score by one point if pacman collide with the Ghost.**

Past your code hpublic void moveGhosts() {

foreach (GameGhost g in game.ghosts) {

if (collider.isGhostCollideWithPacMan(g))

{

game.addScorePoints(-1);

}

g.move(g.nextCell());

}

}ere

Q No 3: add a Horizontal moving Ghost

Past your code hereclass GameGhostHorizontal:GameGhost

{

private GameDirection direction = GameDirection.Right;

public GameGhostHorizontal(Image ghostImage, GameCell startCell): base(ghostImage)

{

base.CurrentCell = startCell;

}

public override void move(GameCell gameCell)

{

if (base.CurrentCell != null)

{

base.CurrentCell.setGameObject(Game.getBlankGameObject());

}

base.CurrentCell = gameCell;

}

public override GameCell nextCell()

{

GameCell gameCell = base.CurrentCell;

GameCell gameCell2 = base.CurrentCell.nextCell(direction);

if (gameCell2 == gameCell)

{

if (direction == GameDirection.Right)

{

direction = GameDirection.Left;

}

else if (direction == GameDirection.Left)

{

direction = GameDirection.Right;

}

}

else

{

gameCell = gameCell2;

}

return gameCell;

}

}

}

**Q No 4: add a ChasingGhost**

Past your code hGameObject pacman;

private GameDirection direction = GameDirection.Right;

public GameGhostChaser(Image ghostImage, GameCell startCell,GameObject pacman) : base(ghostImage)

{

base.CurrentCell = startCell;

this.pacman = pacman;

}

public override void move(GameCell gameCell)

{

if (base.CurrentCell != null)

{

base.CurrentCell.setGameObject(Game.getBlankGameObject());

}

base.CurrentCell = gameCell;

}

public override GameCell nextCell()

{

GameCell gameCell = base.CurrentCell;

GameCell gameCell2 = base.CurrentCell.nextCell(direction);

List<double> distance = new List<double>();

distance.Add(calculateddistance(GameDirection.Left));

distance.Add(calculateddistance(GameDirection.Right));

distance.Add(calculateddistance(GameDirection.Up));

distance.Add(calculateddistance(GameDirection.Down));

if (distance[0] <= distance[1]&& distance[0] <= distance[2]&& distance[0] <= distance[3])

{

direction = GameDirection.Left;

}

else if (distance[1] <= distance[0] && distance[1] <= distance[2] && distance[1] <= distance[3])

{

direction = GameDirection.Right;

}

else if (distance[2] <= distance[0] && distance[2] <= distance[1] && distance[2] <= distance[3])

{

direction = GameDirection.Down;

}

else

{

direction = GameDirection.Up;

}

return gameCell;

}

double calculateddistance(GameDirection direction)

{

return Math.Sqrt(Math.Pow((pacman.CurrentCell.X-this.CurrentCell.nextCell(direction).X),2)+ Math.Pow((pacman.CurrentCell.Y - this.CurrentCell.nextCell(direction).Y),2));

}

}ere