INTRODUCTION TO ROBOTICS PROGRAMMING

Team 294

GOALS

- Introduce FRC style programming with Java
- Learn how to use the development tools
- Practice programming on a virtual robot
- Learn to work as a team
- Prepare for the tryout

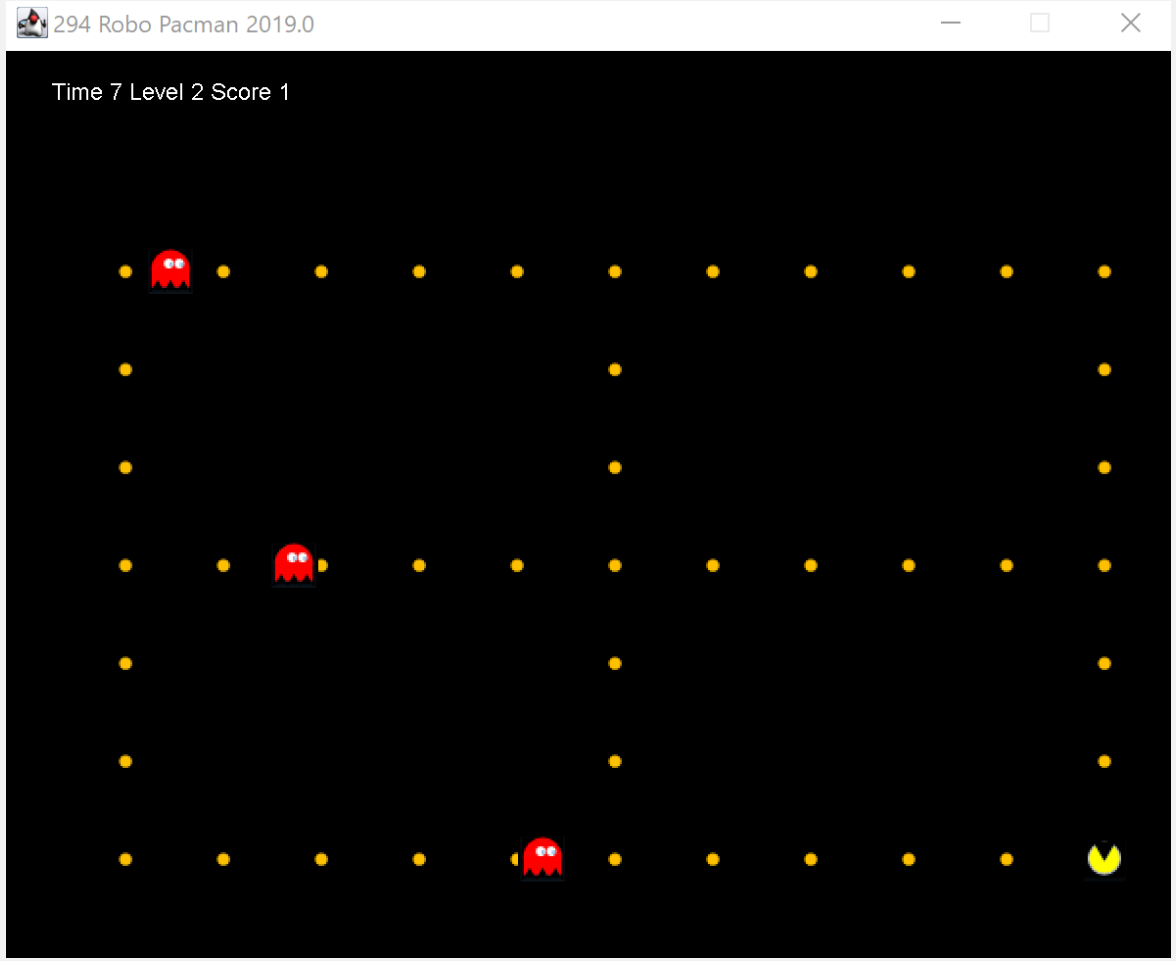
GETTING STARTED

- No prior programming experience is required but...
- Everything will be provided in the lab but you can bring your own laptop or work at home as well
- Work in teams
- New challenge every week
- Not everyone needs to move at the same pace
- Not all the challenges need to be completed

ROBO PACMAN

Robot simulator

- Command based programming
- Autonomous
- Tank drive
- Dot sensor
- Ghost sensor



DRIVETRAIN

Tank drive is one type

- tankDrive(double left, double right)
- int getDistance()
- int getAngle() // 0 is north, 90 east, 180 south

Example: Robot.driveTrain.tankDrive(1, 1)

COMMAND BASED PROGRAMMING

When the Robot runs it executes a series of commands

Command groups control the order

The scheduler repeatedly calls each command until the command is finished and then moves on to the next

Can be executed sequentially or in parallel in the real robot but Pacman only supports sequential

```java
package pacman.commands;

import pacman.base.CommandGroupBase;

public class AutoGroup extends CommandGroupBase {

    public AutoGroup() {
        addSequential(new DriveStraight(200));
        addSequential(new Turn(90));
        addSequential(new DriveStraight(200));
        addSequential(new Turn(0));
        addSequential(new DriveStraight(200));
    }

}
```

COMMANDS

Commands should be reusable and designed to be grouped together to achieve a goal

- Turn
- DriveStraight
- PickupBall

Extends CommandBase

- void initialize()
- void execute()
- boolean isFinished()

```java
package pacman.commands;

import pacman.base.CommandBase;
import pacman.robot.Robot;

public class SpinForever extends CommandBase {

    protected void execute() {
        super.execute();

        System.out.println("Hello world, watch me spin to the right");
        Robot.driveTrain.tankDrive(1, 0);

    }



}
```

TYPICAL SEQUENCE

1. init()
2. execute()
3. isFinished() returns false
4. execute()
5. isFinished() returns false
6. execute()
7. isFinished() returns true

CONSOLE LOG

1. RobotRunner:initialize command pacman.commands.SpinOnce

2. RobotRunner:execute command pacman.commands.SpinOnce
3. tankDrive left:1.0 right:0.0
4. tankDrive right to angle:90
5. tankDrive angle:90 dist:0.0 position:(11,8)

6. RobotRunner: check if command is finished
7. RobotRunner:command pacman.commands.SpinOnce isFinished: false

CONSOLE LOG – SECOND TIME

1. RobotRunner:execute command pacman.commands.SpinOnce

2. tankDrive left:1.0 right:0.0
3. tankDrive right to angle:180
4. tankDrive angle:180 dist:0.0 position:(11,8)

5. RobotRunner: check if command is finished
6. RobotRunner:command pacman.commands.SpinOnce isFinished: false

SETUP

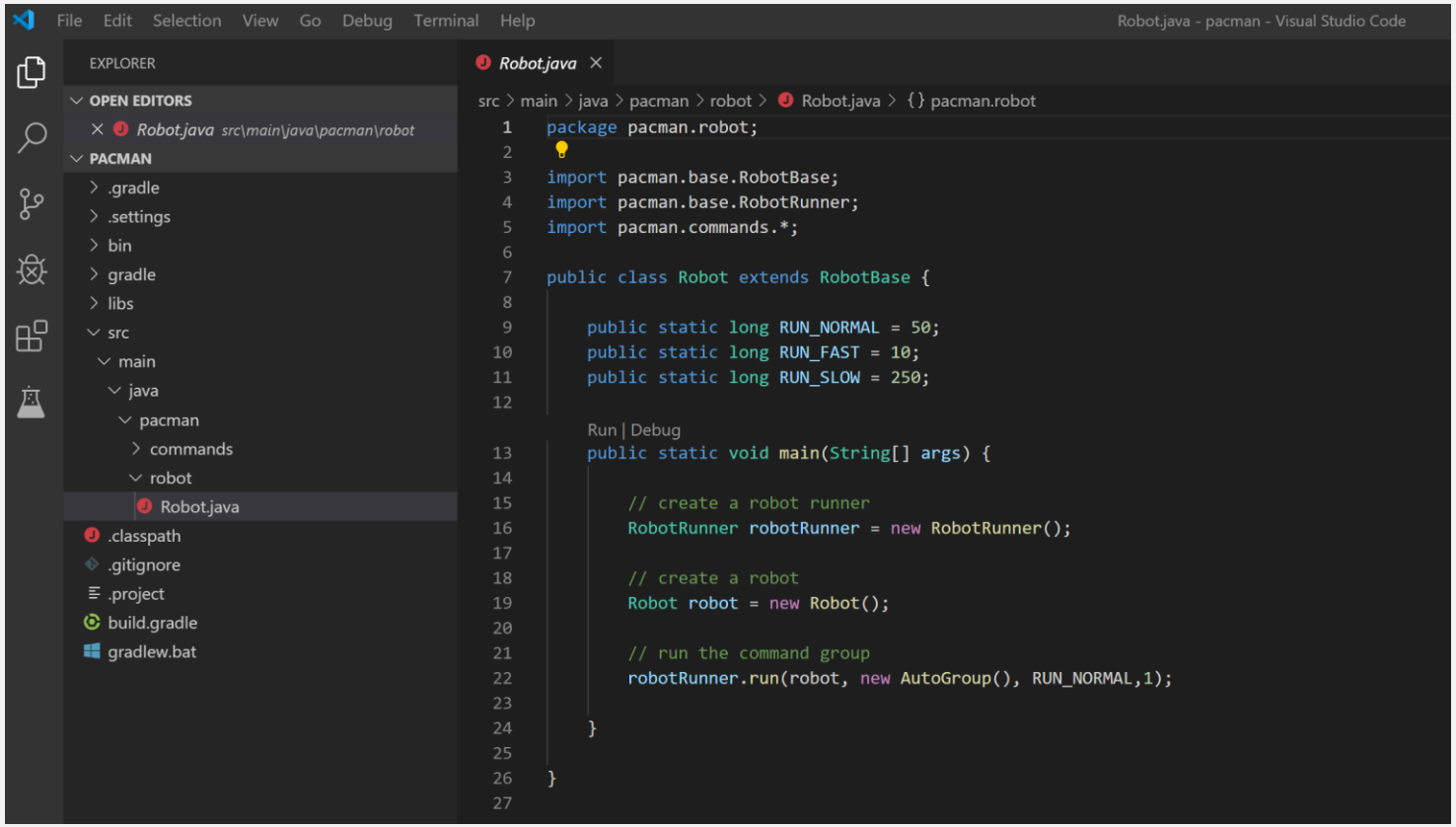
- Windows 10 is used in the lab and during competition
  - MacOS or Linux is also fine for Robo Pacman
- Java Development Kit (JDK SE 11 or higher)
  - https://www.oracle.com/technetwork/java/javase/downloads/index.html
- Visual Studio Code will be used in lab but any IDE will work
  - https://code.visualstudio.com/download

INSTALL

- Get the code from Github
  - Download or clone https://github.com/team294/RoboPacman
  - Extract all from zip file
    - Two main directories
      - Engine – source code for the graphics and the game (just FYI, you don't need to understand this)
      - Pacman – source code for the challenges
- Open folder in Visual Studio Code
  - C:\Users\Paul\Downloads\RoboPacman-master\RoboPacman-master\pacman

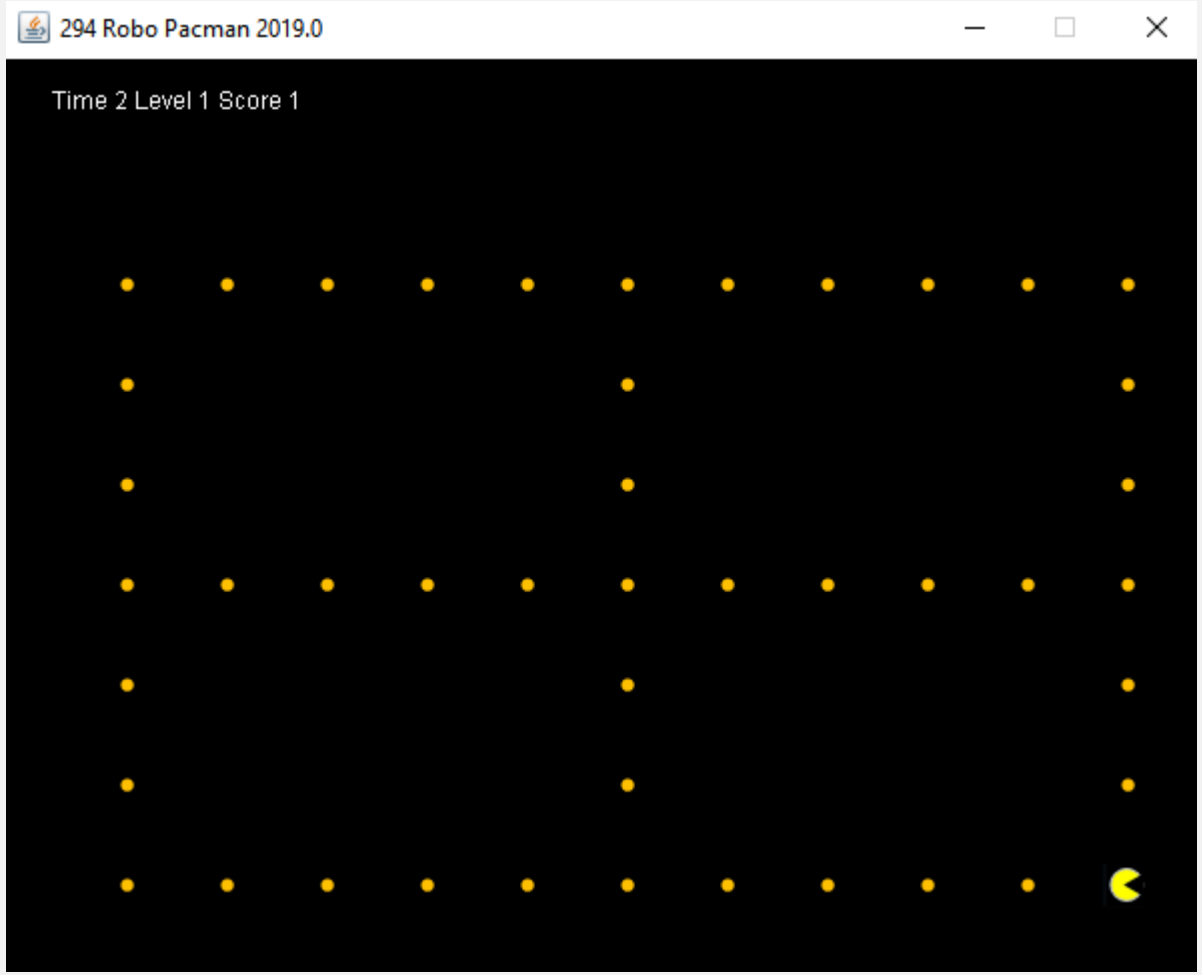
RUN ROBO PACMAN

Run pacman.robot.Robot



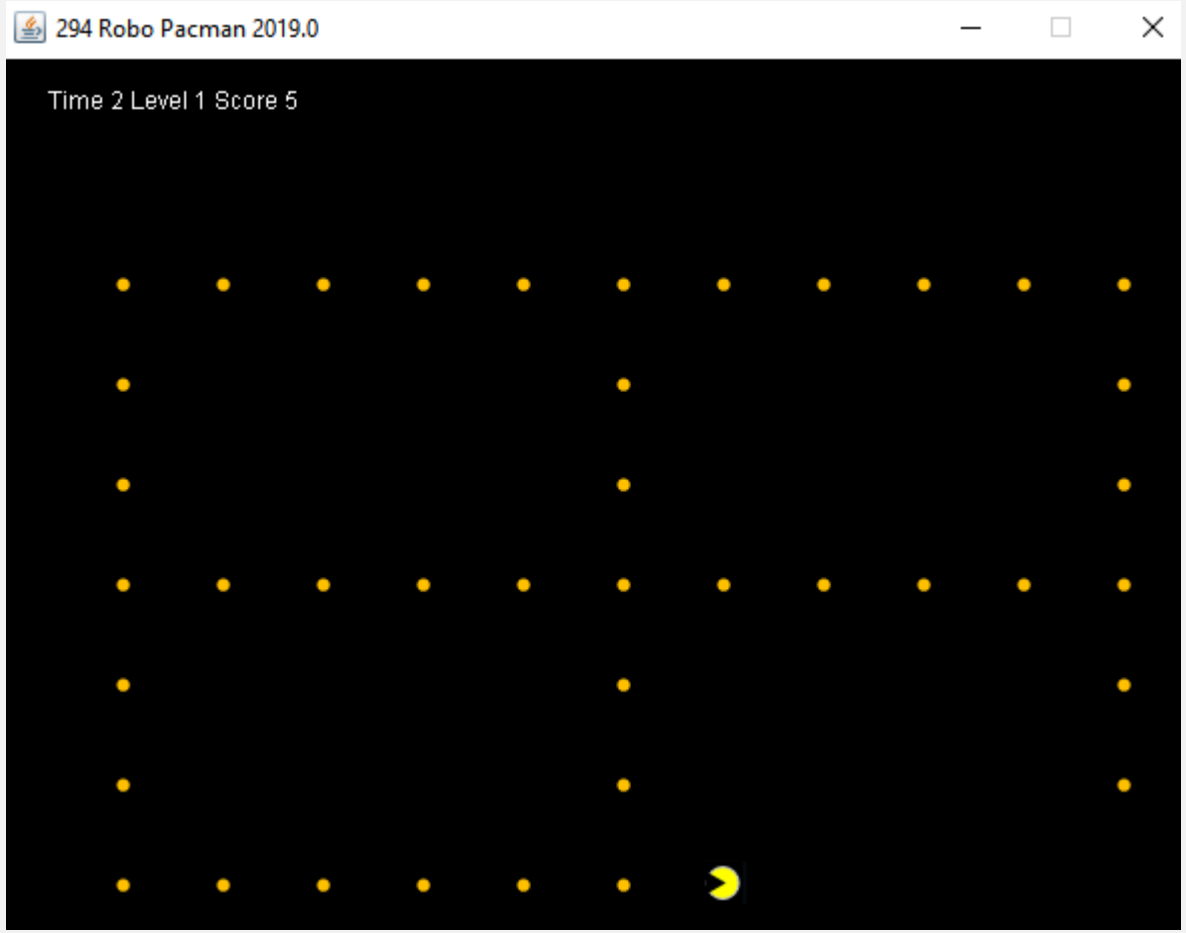
CHALLENGE #1 – SPIN ONCE

- Make Pacman spin 360 degrees and then stop
- Create a new command called SpinOnce
  - Use pacman.commands.SpinForever as an example
- Change AutoGroup to call your new command
- Run Robot on level 1 to test



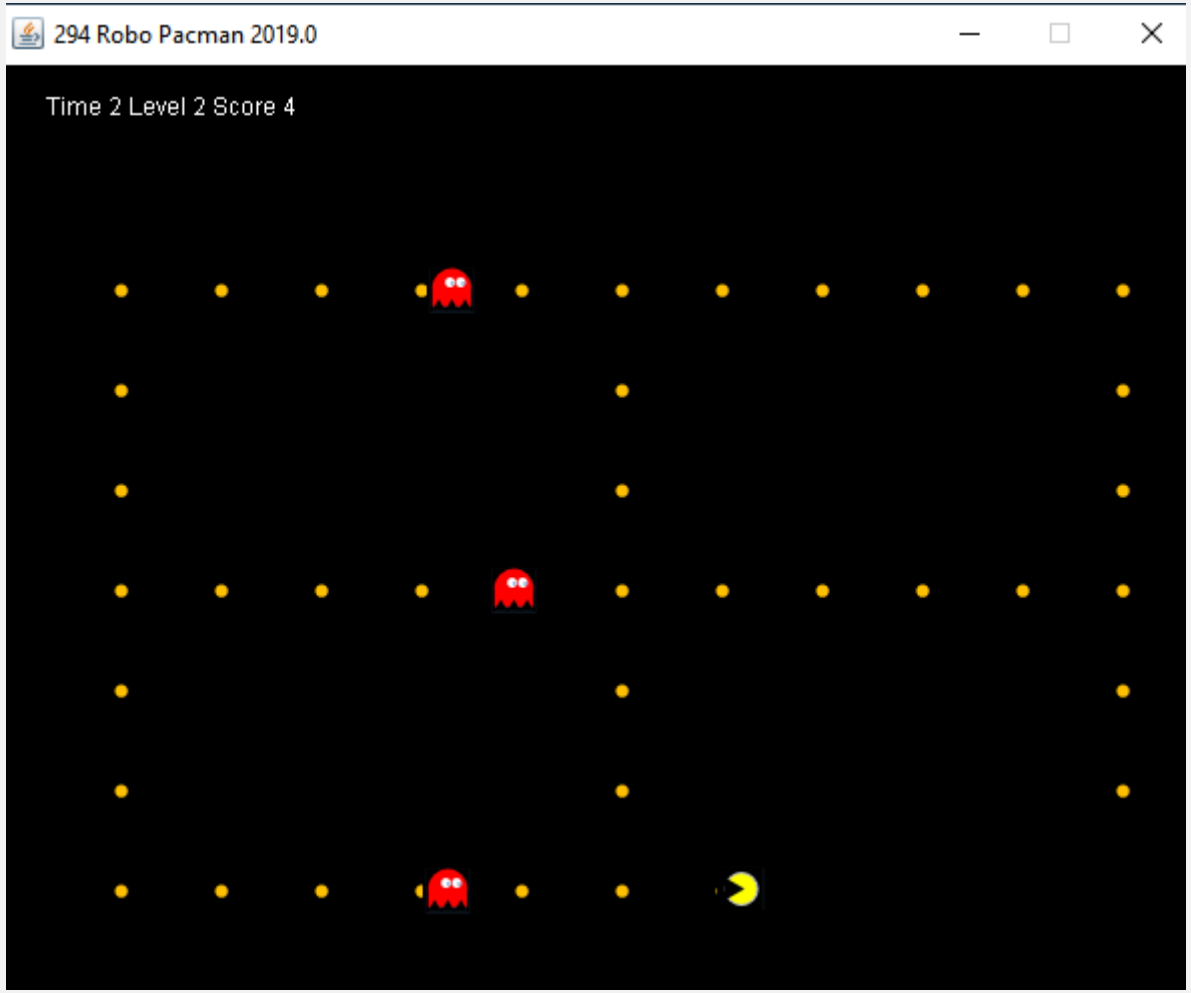
CHALLENGE #2 – EAT THE DOTS

- Make Pacman move around the field and eat all the dots
- Create a new command called EatDots
  - Use pacman.commands.SpinForever as an example
- Change AutoGroup to call your new command
- Run Robot on level 1 to test



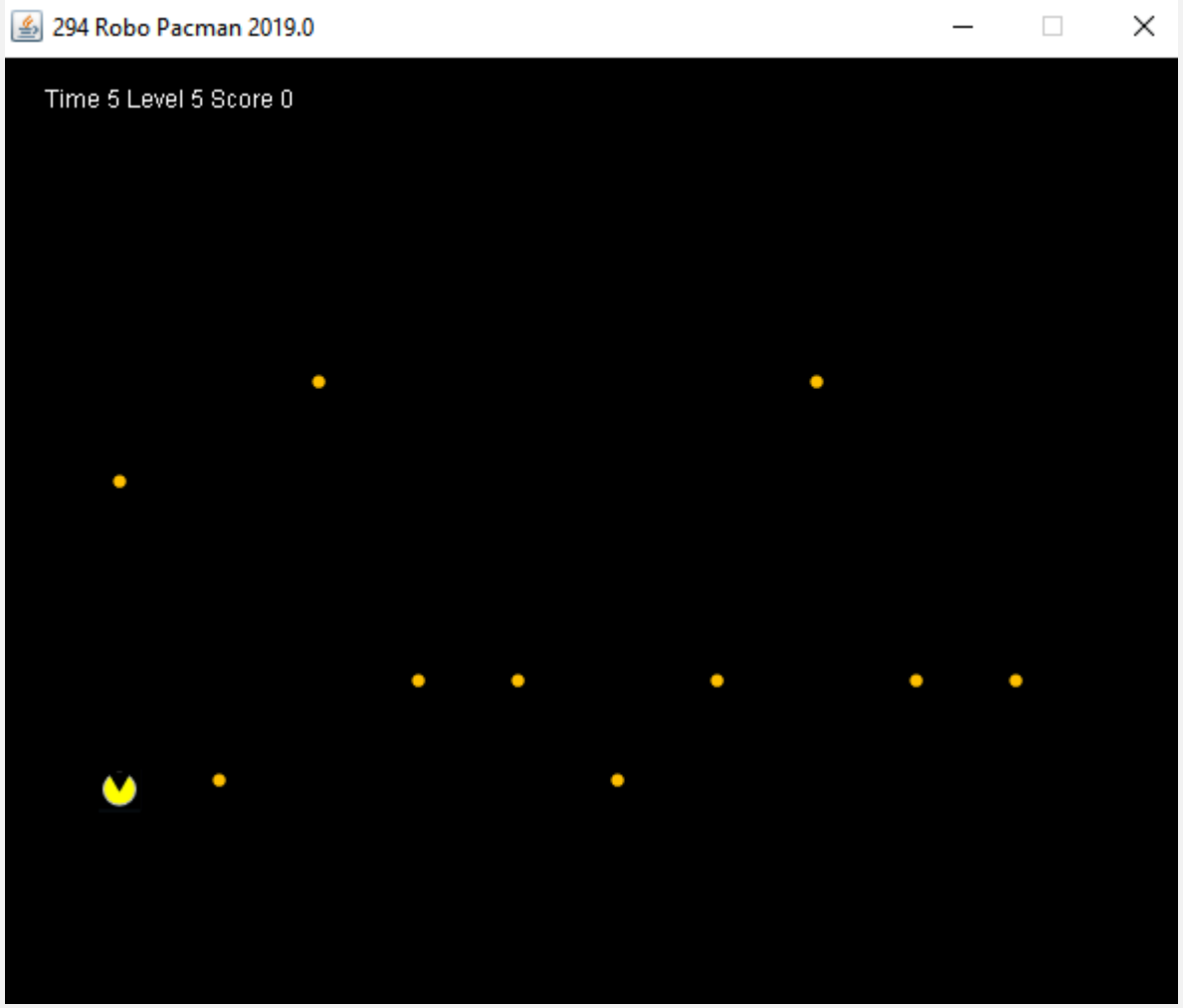
CHALLENGE #3 – AVOID THE GHOSTS

- Make Pacman move around the field and eat all the dots while avoiding the ghosts
- Run Robot on level 3 to test



CHALLENGE #4 – FIND THE DOTS

- Make Pacman find all the randomly placed dots
- Use the DotSensor to find a path to the next dot
- Run Robot on level 5 to test



CHALLENGE #5 – FIND THE DOTS AND AVOID THE GHOSTS

- Make Pacman find all the randomly placed dots but avoid the ghosts
- You will need a path finding algorithm
- Run Robot on level 6 to test