INTPROG: Teaching Block 2

Individual Coursework 2016/17: Pinball Machine

Set: **2nd March 2017**.

Electronic Submission: Submit zipped BlueJ project folder to Moodle by 23.55 on **Friday 24th March**. Detailed instructions are in section 6.

Demonstrations: In your practical class during the final teaching week (27th, 29th, and 31st March 2017).

Worth: 20% of unit marks.

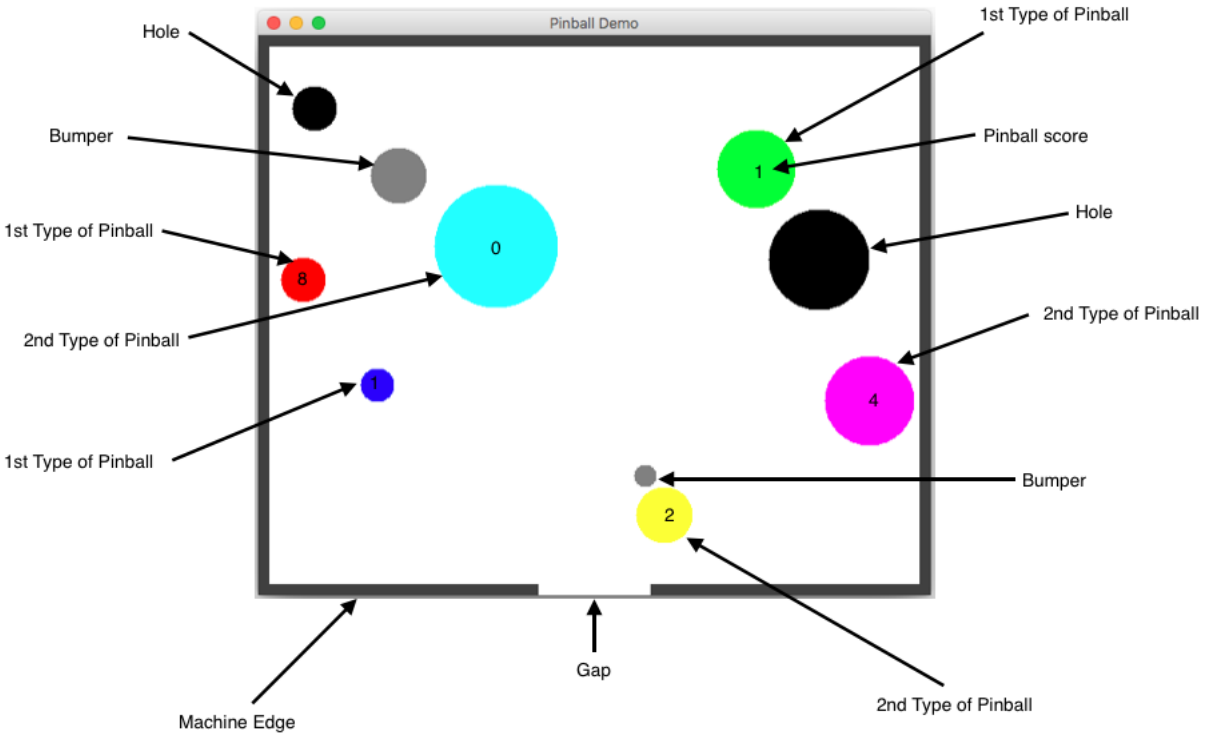
This piece of work is to be done individually. As the assessment will primarily be via a demonstration of your final program it will not be marked anonymously.

1 Aims

To develop object-oriented programming skills, including designing, documenting, testing and debugging of Java code.

2 Description

Your task is to write a program which models a "pinball machine". The pinball model has some significant differences from its real-world counterpart. In particular, multiple pinballs of different types will be used, and there will be no user control (i.e. this is not a game). The machine contains a variety of objects (pinballs, holes, and bumpers), which have different characteristics, the details of which are given below. The simulation will end when one of the pinballs falls through the gap at the bottom of the machine. Pinballs collect points as they interact with particular objects in the machine. Each pinball will have its own score. Once the simulation is over, the final score of all the balls is displayed on the screen. The below example shows a pinball machine with 3 of each type of pinball implemented, 2 bumpers, and 2 holes.



To help you get started a BlueJ project Pinball is on Moodle which provides some of the functionality that the final program will need, but much needs adding. Study this project and understand how the code in the Machine and PinballObject classes work before you start work.

2.1 Objects In The Modelled Pinball Machine

2.1.1 General

All objects are circular and can be any size within a given range. They have a current location (x- and y-coordinates), a size (diameter), a colour (possible colours depend on the type of object, see below), and a speed of travel in both the x and y dimensions (static objects have a speed of 0 for both). The final program will need to include a main loop that processes each object in the pinball machine each time round the loop.

The following sections will provide more details about the specific objects in the machine.

2.1.2 Pinballs

Pinballs move through the machine in straight lines. When they hit the edge of the machine they will bounce off. There are a number of different types of pinball, which all have different behaviours as detailed below. You will be required to implement two different types of pinballs, depending on your student ID number. Your first pinball type may be red, blue, or green. The second pinball type you implement can be cyan, magenta, or yellow.

Each pinball will also store the current score which has been accumulated by that particular ball. Points are collected (or lost) though colliding with the other objects in the machine. The complete scoring system is detailed later in this specification document.

2.1.2.1 Types Of Pinball

Pinballs change some of their properties when they collide with an edge of the machine and also when they hit other balls. There will be no change in behaviour when the pinball collides with a bumper. The five things that may change are:

1. Colour - the pinball will change colour to another valid colour for that type of pinball.
2. Size - the radius of the pinball will increase or decrease by 10%.
3. Speed - the speed of the pinball will change between a given minimum and maximum.
4. Direction - pinballs normally bounce off the edge of the machine, a bumper, or another pinball at the same angle as they hit it at. Pinballs that change their direction will bounce off the object at another random angle.
5. Flashing - the pinball will flash, until it hits another object of the type which causes this action.

The table below lists 10 types of pinball and their behaviour (what property changes) both when they hit the edge of the machine, and when they collide with each other.

Pinball	Edge	Other Pinball
pinball0	Colour	Size
pinball1	Size	Flashing
pinball2	Flashing	Direction
pinball3	Direction	Speed
pinball4	Colour	Direction
pinball5	Direction	Size
pinball6	Flashing	Speed
pinball7	Speed	Colour
pinball8	Size	Speed
pinball9	Colour	Flashing

2.1.2.2 What Types Of Pinball Do You Have To Implement?

You have to write code for 2 different pinball types. The first type of pinball you need to implement corresponds to the final digit of your student number. You also choose one other pinball type that changes two different properties from your first pinball type. There are no marks for implementing the incorrect pinball.

Main Pinball	Other Pinball Options
pinball0	pinball2, pinball3, pinball6
pinball1	pinball3, pinball4, pinball7
pinball2	pinball0, pinball7, pinball8
pinball3	pinball0, pinball1, pinball9
pinball4	pinball1, pinball6, pinball8
pinball5	pinball6, pinball7, pinball9
pinball6	pinball0, pinball4, pinball5
pinball7	pinball1, pinball2, pinball5
pinball8	pinball2, pinball4, pinball9
pinball9	pinball3, pinball5, pinball8

2.1.3 Holes

Holes are static circular objects within the machine, which will always be black. All pinballs lose their points when meeting a hole. If the pinball is the same size or smaller it will fall through and will be lost (i.e. will be removed from the simulation). Otherwise, it will continue to travel around the machine.

2.1.4 Bumpers

Bumpers are static circular objects within the machine, which will always be gray in colour. When a pinball collides with a bumper it will always rebound at the same angle as it hit it at. The pinball will gain 2 points for every collision with a bumper.

2.2 Scoring System

Each time a pinball interacts with an object or wall in the machine it will either gain or lose points. Each pinball will have its own points total, which should be displayed on the ball thoughout the simulation. At the end of the simulation, the total amount of points for all the pinballs must be calculated and displayed on-screen. Points will be awarded for the following interactions:

- Colliding with a wall - 1 point
- Colliding with a bumper - 2 points
- Colliding with another pinball - 5 points
- Coming into contact with a hole - lose all points collected by that pinball

3 Required Functionality

Functionality 1 (30% of marks) - Create a pinball machine that has bumpers and the two types of pinball you are using. You will need to include 3 of each type of pinball. The pinballs will need to have the correct behaviours for colliding with the machine edge and the bumpers, but **not** when 2 pinballs collide. You **will not** need to implement the hole or scoring system for these marks.

Functionality 2 (15% of marks) - Add the functionality for two pinballs colliding with each other, and the associated behaviours.

Functionality 3 (10% of marks) - Add hole objects and their associated functionality.

Functionality 4 (10% of marks) - Add the scoring system.

4 Code Quality

After demonstrating the program's functionality, the member of staff will give you some feedback on the quality of your program code. Your program will be awarded marks based on: its overall structure (ensuring that the principles of object-oriented programming have been incorporated), its readability and structure, and the quality of the algorithms used. Even if your program works well, the code may obtain very few marks if its readability is poor.

5 Documentation

You will be required to produce documentation, using the JavaDoc format of commenting the classes. A brief description should be given for each class, constructor, and method. Both constructors and methods need to provide details about their parameters (where relevant). For methods which are not declared as void, details of the value returned should be provided. For further details, please see practical 6.

6 Submission And Assessment

Please submit via Moodle a zipped file (called upXXXXXX.zip, where XXXXXX is your student ID number) containing all relevant BlueJ project files by 23.55hrs on Friday 24th March 2017. The primary assessment of this work will be via a demonstration, to a member of staff, given in your regular INTPROG practical session between Monday 27th and Friday 31st March 2017. Each demonstration should last approx. 5 minutes. Failure to demonstrate your program will result in a functionality mark of 0% even if you have submitted. Late submissions or demonstrations will be capped at 40% (subject to ECF amendment).

6.1 Marks Breakdown

Functionality 1 - 30%
Functionality 2 - 15%
Functionality 3 - 10%
Functionality 4 - 10%
Code Quality - 25%
Documentation - 10%

6.2 Demonstration Requirements

During your demonstration, you will be given a maximum of 5 minutes to demonstrate your program. You need to plan this demonstration very carefully before submission. If functionality is not clear in the demonstration, you will not be awarded the associated marks. You must demonstrate the following functionality:

Functionality 1

- Presence of bumpers and the 2 correct types of pinball
- Correct behaviour of each type of pinball when hitting the wall
- Correct behaviour of each type of pinball when colliding with a bumper
- Correct ending of the simulation, as described in section 2

Functionality 2

- Correct behaviour of each pinball when colliding with another pinball. (Note: if you have a flashing pinball, you will need to show two collisions - one which starts the flashing behaviour, and another that stops it)

Functionality 3

- Presence of holes
- Correct action when a smaller pinball interacts with a hole
- Correct behaviour when a larger pinball interacts with a hole

Functionality 4

- Presence of the score on the pinballs
- Correct addition of points when colliding with the wall
- Correct addition of points when colliding with a bumper
- Correct addition of points when colliding with another pinball
- Correct points action when interacting with a hole
- Correct display of points at the end of the simulation

It is anticipated that you will have a demo() method, either in the machine class or a separate Demo class. This method cannot contain any functionality for the simulation - it should only create objects and call methods in those objects. Points will be taken off for having behavioural code in the demo() method.

After the demonstration, you should receive an email with your marks. If you don't receive an email, your mark will not have been recorded. It is your responsibility to tell us the email hasn't arrived.

Advice On Completing The Coursework

You may ask me or any of the INTPROG teaching team for any help you may require - you will not be penalized for asking for help. We will not do the coursework for you but will answer any specific problems you have. You may also get help from other students taking the unit but if any submissions have an excessive amount of duplicate code the University's plagiarism procedures will be enforced. In simple terms - what you submit must be your own work not copied from, or done by, someone else.

All code must be professionally written (e.g. with JavaDoc comments and correct indentation) and will be assessed for:

- correctness
- appropriate use of language constructs
- style (commenting, indentation, etc.)

Develop your code in small parts. Test each part as you implement it. Compile your program frequently - don't add more than a handful of program statements without compiling it to check what you've done. Keep backup copies of working code as you complete each task.

Think carefully about the design of each class you create - what attributes and operations should a class offer?

Don't reject well thought out code just because you get compilation errors - work out why you've got the error and correct it, don't throw away the good with the bad.

Don't be over ambitious and do not spend excessive time on this coursework. You should spend approximately 20-30 hours (3-4 full days).

The coursework brief has deliberately not been entirely prescriptive about exactly what you need to do - there is room for individual interpretation. If you have any doubt about what is required or believe there is any ambiguity in the coursework brief please email me with your query and I will post answers to all such queries on Moodle.

Key Points

- Each student must create their correct pinball types as described in this coursework brief.
- You need to start work on this now!
- You will lose code quality marks if you don't use inheritance in your solution.
- Instance variables of classes should never be public. Only constructors and methods may be public. Your assignment will be capped at 40% if you don't follow this standard.
- Providing appropriate JavaDoc comments in your program is all that is needed to get marks for documentation.
- To get a good code quality mark you will need to use ArrayLists or arrays to hold the objects in the machine.
- Failure to attend the demonstration will result in 0 marks for functionality. Late submissions or demonstrations will be capped at 40% (subject to ECF amendment).
- Plan your demonstration carefully in advance so you can demonstrate to the marker all your program functionality in the allocated time (approx. 5 minutes).
- Do not change the Canvas class