SE 3XA3: Test Plan
Namcap

Team 2, VPB Game Studio
Prajvin Jalan (jalanp)
Vatsal Shukla (shuklv2)
Baltej Toor (toorbs)

October 30, 2016

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
2016-10-30	1.0	Addition of content to sections 1.1, 1.2 and 1.4
2016-10-30	1.1	Addition of content to sections 2.1 and 2.2
2016-10-30	1.2	Addition of content to sections 2.3 and 2.4
Date 4	1.3	Notes

This document is the test plan for the Pacman redevelopment project, Namcap. The test plan outlines the testin methodologies and techniques to be used when testing the functionalities and characteristics of the system and its component parts.

1 General Information

1.1 Purpose

The purpose of testing is to ensure that the developed implementation functions correctly and to address any areas where the system is vulnerable. Through the formal specification of the testing methods and verification techniques, testing the implementation becomes mroe reliable.

1.2 Scope

As Namcap is a redevelopment of a classic arcade game, the test plan will aim to formalize the various functionality testing techniques as well as the usability tests utilized in order to ensure that the implementation meets the given requirement. This document will explicitly detail the different methods and testing tools to be utilized for this project.

1.3 Acronyms, Abbreviations, and Symbols

Table 2: **Table of Abbreviations**

Abbreviation	**Definition**
Abbreviation1	Definition1
Abbreviation2	Definition2

Table 3: **Table of Definitions**

Term	**Definition**
Term1	Definition1
Term2	Definition2

1.4 Overview of Document

This document will describe the testing methodologies to be utilized to verify Namcap as an implementation and development. The test plan will outline all testing tools, schedules, automated and manual tests, tests to address the requirements for the application, unit tests, and any additional testing performed on the PoC and existing implementation.

2 Plan

2.1 Software Description

Namcap is a redevelopment of the classic 2D arcade game Pacman. The gameplay involves the player sprite moving through a 2D level attempting to acquire (collide with) as many dots as possible to increase the score. Enemy sprites (ghosts in the original) will move throughout the level and upon collision with the player will cause the player to lose a life and/or end the game. The player can however consume (collide with) a power up to send the enemies back to the center of the level (when collision occurs). The implementation covers these aspects of the core gameplay (scoring, collision, movement, and enemy mechanics).

2.2 Test Team

The test team for Namcap is comprised of Prajvin Jalan, Vatsal Shukla, and Baltej Toor.

2.3 Automated Testing Approach

For the purposes of automated testing, the test team will use both JUnit and the built-in Robot class library. JUnit is a unit testing framework that will run automated tests for most logical components and any GUI components where applicable. The Robot class library will be used to automate testing that simulates user input. Based on the simulated user input, logical and GUI components will be tested to ensure that the appropriate collision and scoring responses occur.

2.4 Testing Tools

The automated unit testing tool JUnit and the Robot class library are the only testing tools that the team will use to verify the implementation.

2.5 Testing Schedule

See Gantt Chart at the following url ...

3 System Test Description

3.1 Tests for Functional Requirements

3.1.1 Area of Testing1

Title for Test

1. test-id1

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

3.1.2 Area of Testing2

...

3.2 Tests for Nonfunctional Requirements

3.2.1 Area of Testing1

Title for Test

1. test-id1

   Type:
   Initial State:
   Input/Condition:
   Output/Result:
   How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.
   Initial State:
   Input:
   Output:
   How test will be performed:

3.2.2 Area of Testing2

...

4 Tests for Proof of Concept

4.1 Area of Testing1

Title for Test

1. test-id1

   Type: Functional, Dynamic, Manual, Static etc.
   Initial State:

Input:

Output:

How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

4.2 Area of Testing2

...

5 Comparison to Existing Implementation

6 Unit Testing Plan

6.1 Unit testing of internal functions

6.2 Unit testing of output files

7 Appendix

This is where you can place additional information.

7.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

7.2 Usability Survey Questions?

This is a section that would be appropriate for some teams.