Black-Box Testing:

Our black-box testing is a mixture of component and system testing as defined by Summerville [1]. Due to the nature of the product, we found it hard to test individual components in this way without a large portion of the system. For example, rendering of sprites can't be done without a screen to render them on so testing that they render and have the correct texture is difficult without having a screen creation and rendering system. As a result, it can be hard to separate these types of testing. We have laid our testing for this section out in a table to make information for each test easily accessible. To improve our testing and make it easier to reference we have given each test an identifier and identified which requirement the test tests. In addition, we included a column explaining the tests, one explaining their results and one explaining the reaction to a test if one was needed. We believe this was a clear layout and contained a good amount of information on each test. Requirements referenced can be found here: https://lloydbanner.github.io/SEPR-Team-7/Req1.pdf.

Non-functional requirements 3.0, 4.0, 5.0 and 7.0 haven't been tested. These requirements are subjective and would require a sample audience to test effectively. Some of these features aren't currently implemented or have very little implementation so they aren't worth testing yet. These will be tested in later assessments when it is more appropriate. These features aren't currently implemented as they weren't required for this assessment and the team didn't have time to finish them for this deadline.

Test ID	Requirements Tested	Test	Result	Reaction
T1.0	**F2.0**	Collisions with power up. Tested by walking into power-ups within the game from various directions and checking that they interacted after a collision.	Left: interacts correctly Right: interacts correctly Top: interacts correctly Bottom: interacts correctly Top-Right: interacts correctly Top-Left: Bottom-Left: interacts correctly Bottom-right: interacts correctly	No reaction needed. The collisions work as required. When the player touches the item it is activated.
T1.1	**F2.0**	Testing power-up has	Health power-up:	Checking why the player was

		required effect after collision.	Heals player more than required Shield power-up: makes the player invincible for 10 seconds as expected. Speed power-up: Speed player up for 10 seconds as expected.	healed more than required. Many health power-ups were stacked on top of each other resulted in the player being healed more. With one power up it works correctly.
T2.0	**F6.0**	Playing the game to check that it runs and the camera is from a 2D top-down perspective.	Running the game it is 2D and all sprites and models are 2D throughout.	None needed. May need to test again after the game has been completed.
T2.1	**F6.1**	Testing that while playing the camera is locked to follow the player.	Player is always in the centre of the screen after movement in any direction or after collisions in any direction.	None needed. May need to test again if new features are implemented that could affect this.
T3.0	**F7.0**	Testing that at any time in the game pressing esc will bring up the pause menu.	Works at any point in the game, apart from after the game is won. When the game is won pressing escape toggles the win screen.	May need to change this as the exit button for the game is on the pause screen. However, the win screen is a paused version of the game. Going forward it might be worth it to reevaluate how this is implemented.
T3.1	**F7.2**	Testing that the controls can be	The controls screen is	Currently works but will need to

		accessed at all times.	available at any time the game can be paused, however, it currently doesn't have the controls on it.	be retested if other controls are added.
T4.0	**F8.0**	Checking that the health bar is visible to the player in all game states.	When playing: visible When paused: visible When health is zero: visible When game is won: visible	None required, health bar and GUI are visible at the required times.
T4.1	**F8.1**	Checking if the player can be attacked when they respawn.	The player isn't usually attacked when they spawn, but they can attract zombies to the spawn location or over time they can wander to spawn.	A way of making sure the player is safe when they spawn needs to be implemented.
T5.0	**F11.0**	Playing the game and testing that the combat acts in real time.	Combat is in real time, you can fight the zombies and they will attack you in real time. Pausing the game can't be used to make this feel turn based effectively.	None needed. Functioned as required.
T6.0	**NF1.0**	Playing the game and checking that areas that resemble the university can be reached.	Buildings and their interiors look like areas from the campuses at the university.	None required.

T7.0	**NF2.0**	Playing through the game and checking that all areas of the game run on a university computer.	All areas run appropriately on a university computer.	None required, will need to retest if new areas are added.
T8.0	**NF6.0**	Playing through the game and checking that it doesn't crash in normal gameplay.	The game doesn't currently crash while playing.	None required, but new features could easily change this.
T9.0	**F1.0**	Playing the game various times and checking there are three player types.	There are currently only two player types, but more will be added in later assessments.	Added a third player type in the next assessment.
T9.1	**F1.1**	Playing the game to see if the Computer Science student player type is available and functions correctly.	This class is currently not available, but will be added in the next assessment.	Add Computer Science Student player type.
T9.2	**F1.2**	Playing the game to see if the Drama student player type is available and functions correctly.	Drama student is available and can disguise as a zombie effectively.	None required.
T9.3	**F1.3**	Playing the game to see if the Sports Player player type is available and functions correctly.	Sports player is available, but only has increased damage and not increased speed. This wasn't added as increased speed made the game to easy	Add a different ability for the sports player to make gameplay more interesting.

			as a sports player.	
T10.0	**F3.0**	Looking for the minigame within the game and checking it functions.	The minigame currently isn't in the game.	This will be added to the game in future assessments. We believed it would be best to wait to implement this as no other part of the game relies on it.
T11.0	**F4.0**	Playing the game and counting the number of locations the player goes through.	3 locations are currently traveled to. These include Heslington West, east and Computer Science.	Other locations will be add in further assessments.
T11.1	**F4.1**	Counting the number of areas with buildings in within the game while playing.	Heslington west and Heslington east areas are in the game with the buildings in these areas. However, only computer Science is accessible.	Other buildings will be accessible in the next assessment as there wasn't time to add more than three locations in this assessment.
T12.0	**F5.0 + F5.1**	Counting bosses and checking they function correctly while playing.	No bosses are currently implemented.	Bosses will be implemented in the next assessment. We left them out of this assessment to prioritize the required implementations.
T13.0	**F9.0**	Playing through the game to see if there is a	There is currently only one zombie	New zombies will be implemented

		variety of enemies and that they function differently.	type.	later, these shouldn't be hard to implement and will just require variations on the same object.
T14.0	**F10.0**	Playing the game and checking if an intro plays at the start.	No intro plays at the moment.	Intro will be a short text section and shouldn't be hard to implement so we decided it wasn't important to add currently.

White box testing-

Our white box tests completely comprised of Junit tests. These allowed us to make sure functionality did not change while we were quickly iterating through versions of our game. As mentioned in the testing introduction LIBGDX did not allow us to make a huge amount of unit tests so we have created what we can with the given architecture.

Test ID	Requirement ID	Test	Result	Reaction
T16.0	**F2.0**	Unit test that checks if health item restores 2 health.	Pass - shown in appendix	None required
T16.1	**F2.0**	Unit test that checks if health item increases health above max health.	""	""
T17.0	**F2.0**	Unit test that checks if speed item increase	""	""

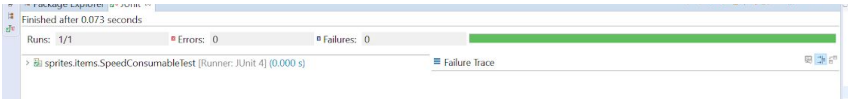
		speed the correct amount.		

References

[1] I. Sommerville , Software Engineering, edition: 9, pp. 216-221, available: https://edisciplinas.usp.br/pluginfile.php/2150022/mod_resource/content/1/1429431793.203Software%20Engineering%20by%20Somerville.pdf

Appendix

SpeedConsumableTest-



HealthItemTest-