5.a Our team’s continuous integration methods will include using tools such as GitHub actions to produce a build of the game every time a change is committed to the project. We will configure it so that it automatically runs tests we have produced every time someone pushes to the main branch of the project. We decided on this method as the group is most familiar with GitHub and we have been using it to code collaboratively throughout the project. The frequent testing will also allow for the group to ensure that no new bugs are created by adding new features or updating old code. The team's use of GitHub actions also will allow members to perform manual tests of the project, allowing the team to evaluate the game's playability as new content is added. This is due to being able to easily access the most recent build of the game as GitHub actions produces a new release of the project each time a change is made.

5.b The team used GitHub actions to automate our continuous integration activities. GitHub actions is a tool provided by GitHub that allows for the automation of continuous integration. It does this by running workflows that can be triggered by an event in the repository, manually run or set to a defined schedule. You can choose to have multiple workflows within a repository, but we only needed to use one to implement our continuous integration methods. Our workflow was triggered by pushing onto the main branch of the repository which allowed for the team to easily access and test a build of the game with all the latest changes implemented. This can be seen in line 47 of the workflow file “gradle-publish.yml”:

```
on:
  push:
    branches:
      - "main"
```

The ‘on’ key is used to trigger the workflow, ‘push’ is the event, ‘branches’ allows you to specify which branch the push needs to be on to trigger the workflow and “main” is referring to the main branch of the project.

We configured our workflow to automatically run a compile test. If the test passes a new release is published as a .jar file to our repository. On the front page of the repository, there is also a badge that displays whether the current release is passing or failing.

The code for the compile test is shown from line 67 of the workflow file:

```
name: "Build with Gradle"
        uses: gradle/gradle-build-action@937999e9cc2425eddc7fd62d1053baf041147db7
        with:
          arguments: desktop:dist
```

This builds the desktop-1.0.jar file and ensures that it compiles correctly. If this part of the workflow fails, the new build will not be published.