Name: Bryan Frahm Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

I will be in charge of Level Design. I will integrate a procedurally generated world, that will have multiple room types that the player will have to navigate.

## Use case diagram with scenario \_\_14

[Use the lecture notes in class.

Ensure you have at least one exception case, and that the <<extend>> matches up with the Exceptions in your scenario, and the Exception step matches your Basic Sequence step.

Also include an <<include>> that is a suitable candidate for dynamic binding]

Example:

### Use Case Diagrams



### Scenarios

**[You will need a scenario for each use case]**

**Name:** Add Numbers

**Summary:** The accountant uses the machine to calculate the sum of two numbers.

**Actors:** Accountant.

**Preconditions:** Calculator has been initialized.

**Basic sequence:**

**Step 1:** Accept input of first number.

**Step 2:** Continue to accept numbers until [calculate] is entered.

**Step 3:** Accept calculate command.

**Step 4:** Calculate and show result.

**Exceptions:**

**Step 1:** [calculate] is pressed before any input: Display 0.

**Step 2:** A button other than [calculate] or a number input is pressed: ignore input.

**Post conditions:** Calculated value is displayed.

**Priority:** 2\*

**ID:** C01

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

## Data Flow diagram(s) from Level 0 to process description for your feature \_\_\_\_\_\_\_14

[Get the Level 0 from your team. Highlight the path to your feature]

Example:

### Data Flow Diagrams



### Process Descriptions

Assign rooms\*:

WHILE teacher in two places at once OR two classes in the same room

Randomly redistribute classes

END WHILE

**\*Notes**: Yours should be much longer. You could use a decision tree or decision table instead if it is more appropriate.

## Acceptance Tests \_\_\_\_\_\_\_\_9

[Describe the inputs and outputs of the tests you will run. Ensure you cover all the boundary cases.]

**Example for random number generator feature**

Run feature 1000 times sending output to a file.

The output file will have the following characteristics:

* Max number: 9
* Min number: 0
* Each digit between 0 and 9 appears at least 50 times
* No digit between 0 and 9 appears more than 300 times
* Consider each set of 10 consecutive outputs as a substring of the entire output. No substring may appear more than 3 times.

**Example for divide feature**

|  |  |  |  |
| --- | --- | --- | --- |
| Output | Numerator  (int) | Denominator  (int) | Notes |
| 0.5 | 1 | 2 |  |
| 0.5 | 2 | 3 | We only have 1 bit precision for outputs. Round all values to the nearest .5 |
| 0.0 | 1 | 4 | At the 0.25 mark always round to the nearest whole integer |
| 1.0 | 3 | 4 | At the 0.75 mark always round to the nearest whole integer |
| 255.5 | 5 | 0 | On divide by 0, do not flag an error. Simply return our MAX\_VAL which is 255.5. |

## Timeline \_\_\_\_\_\_\_\_\_/10

[Figure out the tasks required to complete your feature]

Example:

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PWks) | Predecessor Task(s) |
| 1. Requirements Collection | 5 | - |
| 2. Screen Design | 6 | 1 |
| 3. Report Design | 6 | 1 |
| 4. Database Construction | 2 | 2, 3 |
| 5. User Documentation | 6 | 4 |
| 6. Programming | 5 | 4 |
| 7. Testing | 3 | 6 |
| 8. Installation | 1 | 5, 7 |

### Pert diagram



### Gantt timeline

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |  | 3 |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 6 |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 7 |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |