Name **Benton Wilson** Mark ______________________/50

1. Brief introduction __/3

My feature is going to be the shooting mechanics for the players. This will entail choosing our shooting method which is going to be a click to shoot where bullets travel in a line to where the mouse/crosshair is pointing. This is also going to have to interact with the health of other objects to deal damage when needed.

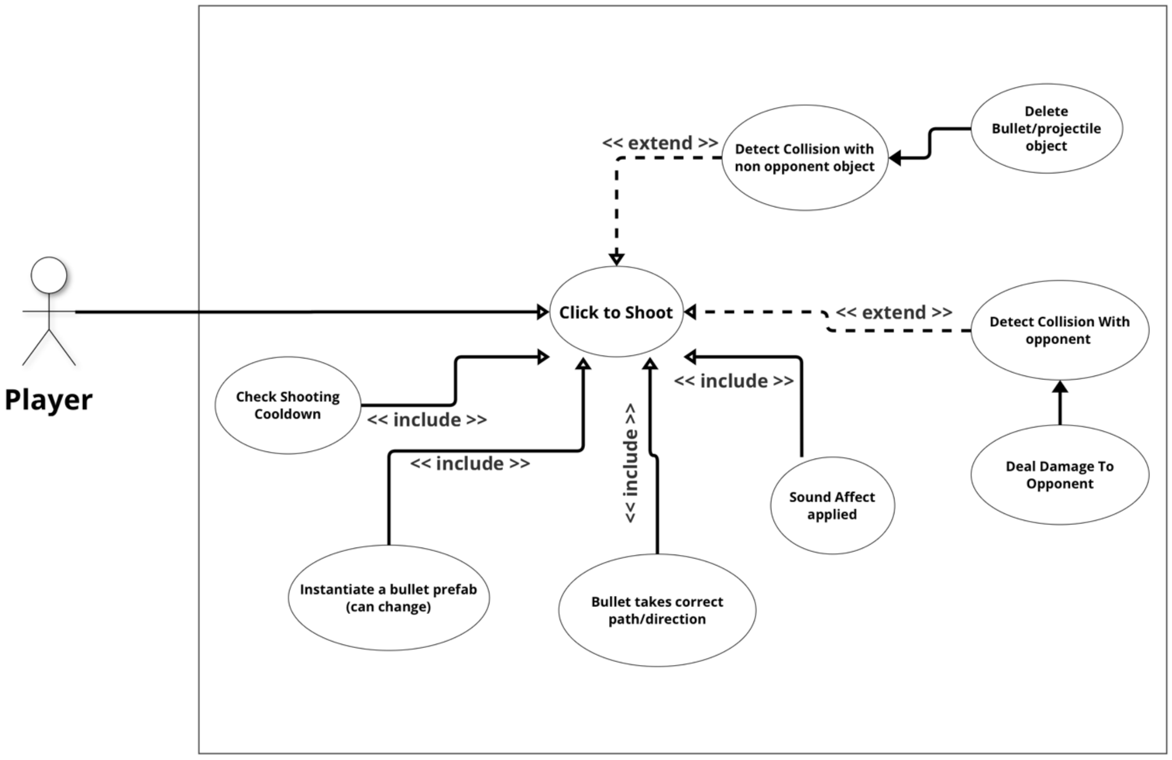
2. Use case diagram with scenario __14

[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]

Use Case Diagrams


Player
Click to Shoot
<< extend >>
Detect Collision with non opponent object
Delete Bullet/projectile object
<< extend >>
Detect Collision With opponent
Deal Damage To Opponent
Check Shooting Cooldown
<< include >>
<< include >>
Instantiate a bullet prefab (can change)
<< include >>
Bullet takes correct path/direction
<< include >>
Sound Affect applied

Scenarios

Name: Shooting Mechanic

Summary: This will deal with how a player shoots and deals damage to items such as opponents.

Actors: Player

Preconditions: Objects needed to be used (bullets/projectiles) have been made and can be used. Health of other objects is initialized and can be manipulated.

Basic sequence:

Step 1: The player or opponent initiates a shooting mechanic. (Click)
Step 2: Check If cooldown on shooting.
Step 3: A bullet prefab is made
Step 4: Bullet takes correct direction (mouse)
Step 5: Sound affect is applied to launching of bullet.
Step 6: Detect collision with some object.
Step 7: If collides with opponent then deal damage else destroy bullet.

Exceptions:

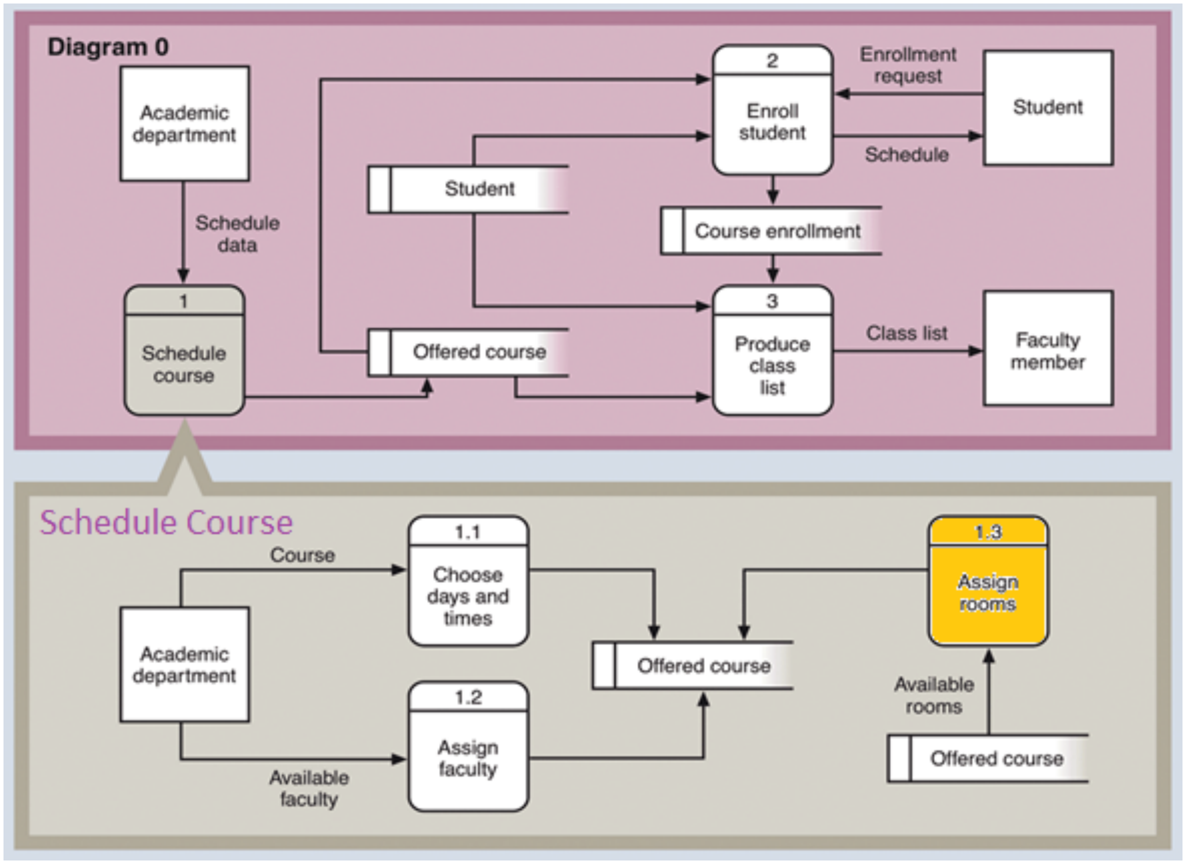
Step 1: Possible different bullet types with different damage
Step 2: A button other than left click used then ignore the input.
Step 3: Don’t allow firing if cooldown is applied

Post conditions: A bullet/projectile has been fired in the correct direction.
Priority: 2*
ID: C01
*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

3. Data Flow diagram(s) from Level 0 to process description for your feature ______14

Data Flow Diagrams

STILL IN WORK



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.

4. Acceptance Tests _______9

Make sure that the collision detection works between a projectile and a object on screen or some opponent object.

Example for projectile collision

Run feature with two objects in scene, an ordinary “obstacle object” and an “opponent” object. Both should be tagged correctly.

<u>Inputs</u>:

- A projectile object on screen
- Movement towards specific objects in game
- Some assigned damage value (Ex. 10)

- Opponent with health (tagged) (Ex. 100)
- Non-Opponent object (tagged)

<u>Outputs:</u>

- Collides with opponent:
    - Damage dealt
    - Projectile destroyed
- Collides with non-opponent:
    - Projectile destroyed

Example for collision detection

Test Case	Description	Input	Output
1	Projectile prefab collides with an opponent tagged object	Projectile Prefab Opponent Object Non-Opponent object	Deal damage to opponent and then should be destroyed.
2	Projectile prefab collides with a **non**-opponent tagged object	Projectile Prefab Opponent Object Non-Opponent object	Should just have the projectile be destroyed immediately.
3	Projectile misses all objects in scene	Projectile Prefab Opponent Object Non-Opponent object	There should be no course of action taken. (This should not happen in game as there will be an outer "boundary")

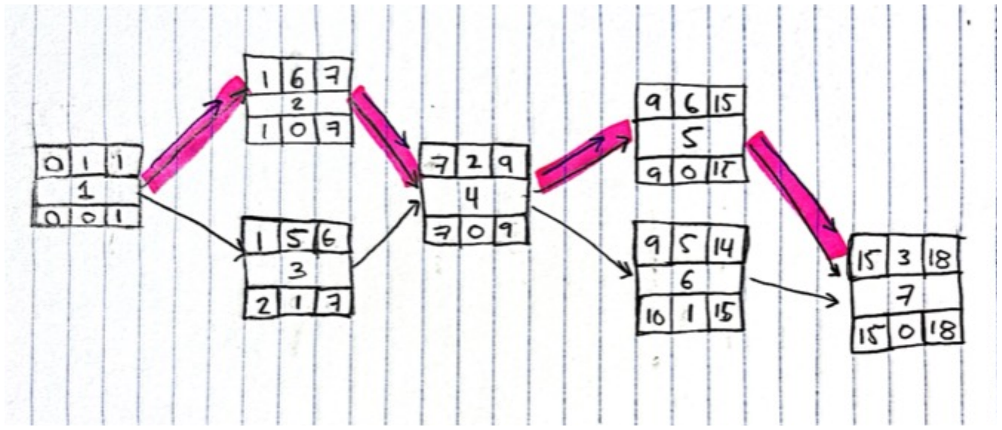
5. Timeline ________/10

Work items

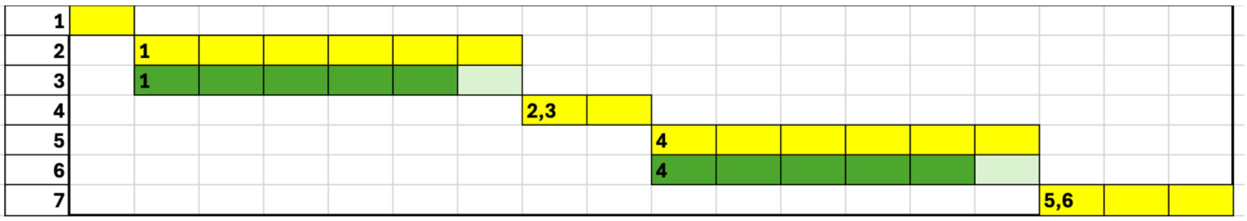
Task	Duration (PWks)	Predecessor Task(s)
1 Requirement Definition	1	-
2. Screen Design	6	1
3. Object Design	5	1
4. User Documentation	2	2,3
5. Programming	6	4
6. Testing	5	4

7. Deployment	3	5,6

Pert diagram



Gantt timeline


1
2
3
4
5
6
7
1
1
2,3
4
4
5,6