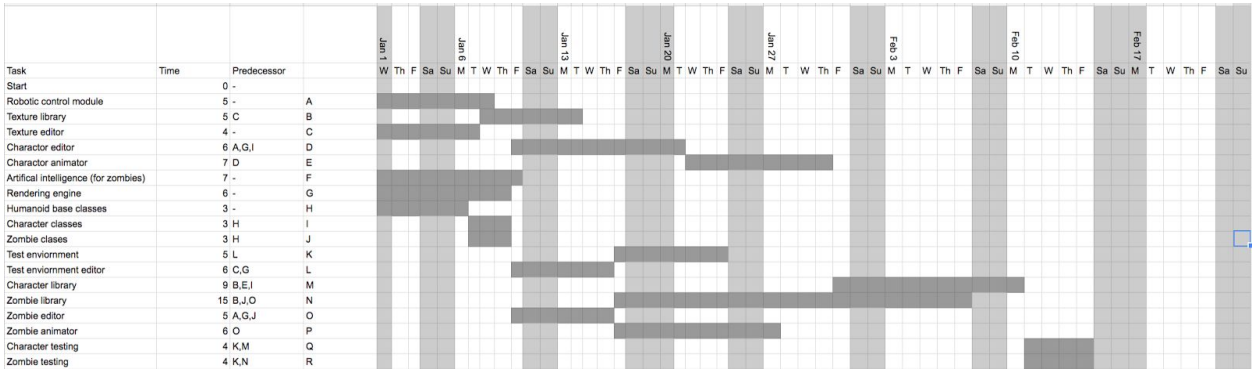
Jay Patel

Homework 1

1. All software engineering projects must handle requirements gathering, high-Level design, low-level design development, testing , deployment, maintenance, and wrap-up.
2. A one sentence description of each:
    a. Requirements Gathering - gather the required information to get ones project going, so what the creator wants to do and what the customers want.
    b. High-Level Design - A large overview of the project and how the major components interact.
    c. Low-Level Design - Break the large pieces into smaller pieces for the coders to then program which will then be put together to make a full picture.
    d. Development - Developing the product (coding).
    e. Testing - Look for errors in the product or project and make sure that everything works as intended.
    f. Deployment - Actually allowing users to use the product.
    g. Maintenance - Keeping the product running and secure as well as updated versions of the product.
    h. Wrap-up - To reflect on the application and learn from it.
3. I did as instructed.
4. JBGE stands for Just Barely Good Enough. It's the philosophy that you shouldn't write any more code documentation or comments than absolutely necessary.
5. The critical path from start to finish goes start -> H -> I -> D -> E -> M -> Q and the total days it will take is 33.
6. 

Task	Time	Predecessor	
Start	0	-	
Robotic control module	5	-	A
Texture library	5	C	B
Texture editor	4	-	C
Charactor editor	6	A,G,I	D
Charactor animator	7	D	E
Artifical intelligence (for zombies)	7	-	F
Rendering engine	6	-	G
Humanoid base classes	3	-	H
Character classes	3	H	I
Zombie clases	3	H	J
Test enviornment	5	L	K
Test enviornment editor	6	C,G	L
Character library	9	B,E,I	M
Zombie library	15	B,J,O	N
Zombie editor	5	A,G,J	O
Zombie animator	6	O	P
Character testing	4	K,M	Q
Zombie testing	4	K,N	R

Just in case this is too small to read I shared the google sheets with you.

7. You can treat deus ex machina problems the same way you handle unexpected sick leave. Add tasks at the end of the schedule to account for completely unexpected problems. When one of these problems does occur, insert its lost time into the schedule.
8. The biggest mistake you can make while tracking tasks is not taking action when a task slips. At a minimum, you need to pay closer attention to the task so that you can take action if it's in trouble. The second biggest mistake is piling more people on the task and assuming they can cut the total time. Unless the new people have particularly useful expertise, bringing them up to speed may make the task take even longer.

9. Five characteristics of good requirements are clear, unambiguous, consistent, prioritized, and verifiable.
10. Listed audience oriented categories:
    a. Business
    b. User/Functional
    c. User/Functional
    d. User/Functional
    e. Nonfunctional
    f. Nonfunctional
    g. Nonfunctional
    h. Nonfunctional
    i. Nonfunctional
    j. Functional
    k. Functional
    l. User/Functional
    m. User/Functional
    n. User/Functional
    o. User/Functional
    p. User/Functional

    Implementation requirements is empty. One might need to buy new hardware or network bandwidth to support the application, but one presumably performing uploads and downloads now, so one may already have everything one needs. In that case, there are no implementation requirements.
11. A list of the some of the things one could do using the moscow technique
    a. Advertising (M)—A phone application typically costs money to install or displays advertising. Currently, the program does neither. It could be modified to display advertising.
    b. Scoring (S)—Right now you either win or lose. The program could be changed to calculate a score.
    c. Quick win (C)—The program could allow the user to type a guess for the whole word to get extra points.
    d. Report high score (W)—The program could let users report their high scores to a central database so that other users can view them on a web page.