Slide 1: FULL NAMES – Joe, Savanah, Matt

Slide 2: Problem Statement – Savanah (explain the game objective)

Slide 3: Controls – Joe

Slide 4: Project Management – Matt (Microsoft Project Gantt Chart that helped keep focused on what needed doing)

Slide 5: Work Breakdown – explain roles (Joe then Savanah then Matt) then last Joe (everyone worked on the images)

Slide 6: Game Structure – Matt – Flowchart showing class breakdown – arrows show inheritance – Tanks object super class

Slide 7: We aren’t graphics Designers – Savanah - Explain the different images  
(from top left to right then bottom left to right)  
Player tank  
Enemy tank  
Mines  
Walls  
Pointer (didn’t get to use…explained later)  
Floor (tileable)

Slide 8: Bullet Mechanics – Joe – explain bullet travel  
 \*\* If you need help \*\*  
 If you have a circle of a specified radius around the center of the tank, you can calculate the angle between the line formed by the mouse-click and the horizontal line, and use trigonometry to determine where this line intersects the circle. Can then use this point to determine step distances for bullet.

Slide 9: Code accompanies slide 8

Slide 10: Set backs – Joe – Collision Detection (many redesigns); Rotating images (difficult and time consuming to figure out)  
 Matt – Readfile (filename after opening file – compiler doesn’t catch); custom pointer (w/o reducing speed or using a graphics API, windows.h, breaks readfile, breaks world drawing); template arrays (can’t return from functions, made a work-around)

Slide 11: Learned – Matt (I hate making games; how collision detection works and logic; planning ahead mostly helped, but issues drove off course; ambiguous errors)

Slides 12 – 15: Screenshots – Savanah – 12 shows gameplay; 13 (Left) player wins…shot tank   
(right) Player loses… shot by ai; 14 (left) Player can place a mine (right) ai dies when hit mine – player wins; 15 Game is pause-able. Everything stops and all that can be done is un-pause or restart

Slide 16: More time – Matt - We were a little ambitious in our plans  
Animations (explosions)  
Sound effects (explosions, collisions)  
More levels (different amounts of enemies and configurations)  
More Sophisticated AI (different types of enemies that work differently)  
Multiplayer  
Scoring

17: Demo – if time permits

18: recap – Joe: Overall, we have a playable game with double-buffered graphics, collision detection, and a fully functioning Artificial Intelligence enemy. This shows our ability as programmers, and we are very satisfied with our progress and outcome.

19: questions? – hopefully not…