Slide 1: Maksim	[TITLE]
Slide 2: Maksim	[TABLE OF CONTENTS]
Slide 3: Zaman	[TITLE] - 1 Introduction
Slide 4: Zaman	Howitzer company is asking for a simulation, to teach its soldiers about kinematics and dynamics What is needed? - Barrel pose (position and orientation), radius and mass of the projectile, drag coefficient, initial speed, force applied to the projectile, and gravity
Slide 5: Zaman	[TITLE] 2 - Problem Definition
Slide 6: Zaman	What is the problem you are trying to solve? Write a program that simulates a Howitzer in the programming language of Java. We want an ACCURATE solution which soldiers are able to use with various inputs to teach them about vertical and horizontal barrel positioning in relation to various external factors such as gravity, wind, coriolis effect, self-propelled rounds and so on.
Slide 7: Maksim	[TITLE] 3 - Design Requirements
Slide 8: Maksim	We modeled our solutions after the M777 Howitzer; it has a barrel length of 5.08m, a shell diameter of 15.5 cm, a shell height of 60cm, and a shell weight of 43.2kg. The maximum firing range of the M777 Howitzer is 24.7 km when shot at a 30 degree vertical angle. If we simulate a shot at a 30 degree angle in our solution it returns 24.8km, which is off by about 100m, but after reading further sources the M777 maximum range could be higher but is rated as lower to guarantee each gun can achieve it.
Slide 9: Maksim	So, getting more technical there are 3 major portions in us planning our design: they are the functions, objectives, and the constraints of it.
Slide 10: Maksim	For functions, we want the ability to request the following from the user: Howitzer input, we want our users to be able to decide the direction of

	fire. Shell inputs, we want our solutions to be able to simulate various shells since there are different shells for different functions which have different characteristics. External forces, the real-world has many external forces such as wind and earth rotation that need to be set and modeled. By default - our main function uses solution 3 which is the most complex solution with 5 forces; we have a selector function to easily change between solutions.
Slide 11: Maksim	As for objectives, we have many but these were the major ones: The Howitzer simulation should model behavior as best as possible; with correct speed, weight, angles of rotation, barrel height and so on - we want this to be almost an equivalent to the real world. The program's physics logic should simulate in a 3D plane, with 3 dimensions of measure; and it should be able to be “seen/tracked” per second. We should have the ability to input different values for the variables to get different results; to simulate a variety of different scenarios. Just to preface, this is a differential equation problem, therefore` to get an accurate simulation, there should be a form of differential calculus being used, for our solution we decided to use a modified version of euler’s equation which is basically a method of approximation.
Slide 12: Zaman	Constraints Economic Factors - Cost of project: This **project does not have a budget** therefore we are using all open source software. There is a cost in the sense of labor required by project contributors as well as any other supplies used by contributors to complete the project; but this will not be realized. - Cost of Use: The equipment that will be used for development will include personal computers along with lab computers; both of these have already been purchased and do not warrant a cost of use; our repo location (GitLab) is also provided free of charge; this unfortunately results in a **lack of DevOps features.** -

	Social Factors - Time is a constraint, scope of project should be kept accomplishable Reliability - Project will have continuous development over a short term, therefor not making it viable long term Sustainability and Environmental Factors - Project must strive to use sustainable alternatives where possible Societal Impacts - Project should have minimal societal impact, other than to train soldiers or other required parties about the howitzer cannon
Slide 13: Simran	[TITLE] 4 - Solutions
Slide 14: Simran	Ok now onto the solutions we created for this project We created a Howitzer Simulator that utilizes Euler's approximation to calculate position as well as taking gravity, acceleration, drag force and coriolis effect into account Overall our first solution was pretty accurate partly due to Euler's approximation and the drag force, without drag force the projections were off by a factor of 1.75, when we added drag the projections became very accurate Now for a demo of solution 1 by max
Slide 15: Simran	[DEMO VIDEO OF SOLUTION 1]
Slide 16: Simran	For the second solution it was same as solution 1 expect we accounted for any external wind velocity. The wind can have a significant impact on the final position of the projectile as can be seen in the demo coming up This solution was more accurate as it added environment variables to our simulation, before other than coriolis we were not accounting for any major environmental variables, this will help control for the wether aspect of the simulation. Now for the demo of solution 2
Slide 17: Simran	[DEMO VIDEO OF SOLUTION 2]
Slide 18: Simran	The final solution was built on the last one, we added self propelled velocity as many shells have a their owm propulsion systems which allows them to go further, this kept the accuracy in the numbers the same

	but improved the real life accuracy as in a real life situation there is a good chance they will be using different types of shells, also why we included changeable shell parameters in the first solution. Now for a demo of the third solution
Slide 19: Simran	[DEMO VIDEO OF SOLUTION 3]
Slide 20: Simran	Now to talk about the testing aspect of this project Let's start with unit testing The first unit tests we did were boundary value analysis, this is testing inputs on their domains. We used robustness testing on 6 inputs so we had 7 tests per input which we felt like was a good number to thoroughly test the inputs Next was equivalence class testing, we took the domains from boundary value analysis such as for z position we had negative infinity to zero as invalid and zero to infinity as valid, we did this for each input and created a table, we then abstracted those domains into test cases which helps simplify functional testing Path testing was were we tested to make sure every single path in our function was covered. For example if an if loop had four conditions we made sure to check all four conditions, we used a combination of a graph and ecl emma which we used in the lab to make sure the whole function was covered.
Slide 21:Simran	[DATA FLOW DIAGRAM] We also created a data flow graph, as you can see on the screen, for our data flow testing, we used def-use coverage to make sure we were hitting all the nodes for the function.
Slide 22:Simran	For Decision Table Testing we created a decision table for our test requirements to make sure none of the input combinations were throwing errors, which was pretty simple For Integration Testing we tested the landing position calculator function and the drag acceleration function which is called by the previous function within the Force Simulator Class. We tested using test stubs and test drivers to check the call sites and then used big bang integration to check overall integration
Slide 23: Simran	[FINITE STATE MACHINE]

	For system testing we created a rough Finite state machine and made sure we had edge coverage which really meant we had complete runs of all of our different solutions
Slide 24: Zaman	[TITLE] 5 - Project Management
Slide 25:Zaman	Gantt Chart We broke the tasks into 4 different big sectors, where these 4 categories have subcategories listed within them, when we started the project, we started by making a group chat to communicate any changes that were made, the gitlab, along with how to onboard to the development. Task 2 was divided into 3 different sectors, as you can see solution 1 through 3, each are their own separate task Task 3 was the testing phase which we then broke into Task 4 was documenting everything, this includes commenting of code, the readme files, presentation, and videos
Slide 26: Zaman	[TITLE] 6 - Conclusion and Future Scope
Slide 27: Zaman	In conclusion, we learned a lot about the testing phases of development from start to finish about a program, from planning, developing, testing, documenting, overall giving us a better understanding of how a real world application is developed and especially how important the testing phases are, because if you test something and it doesn't work, you know it needs to be improved from more aspects, especially with the weekly meetings, which taught to plan out appropriately instead of procrastinating Future Scope As far as our future scope goes, the first implementation would be to create a Graphical User Interface, where it would be more appealing for the user to look at, as opposed to looking at it from eclipse. The second implementation we would make, is a 3D firing visualization, which shows the cannon firing itself, to which then we would have a shell pov, where the shell is shown in the air after being fired, as well as when it lands
Slide 28: Zaman	[END] Thanks