Anticipated Milestones

Milestone #	Description	Completion Date
1	Research and get comfortable with various GUI API's and write a basic application in each of them. Decide how the backbone is going to interract with the GUI.	TBD
2	Create a basic debugger application that does at least what are considered the fundamentals: It contains a text viewer; it has basic buttons for basic debugger functions and interractions; it can attach to a running process, select an .exe to debug; it can load source code into the text viewer; and it can set and manage breakpoints which will also involve the capability of being able to read and write the attached program's CPU registers and memory.	TBD
3	Test and iron out many of the bugs that are likely to occur as a result of Milestone 2 and then enhance the basic application with the nice-to-have features such as catching and interpreting program exceptions, the implementation of a command-line interface, and implementation of the generalized platform layer to encapsulate all of the operating system interractions.	TBD
4	Continue to iron out bugs from Milestone 3 and maybe 2 again and implement the additional remote-control features and potentially attempt to implement the debugger in other languages to enhance portability.	TBD
5	Test again for additional bugs in milestones 2,3, and 4, assure that many issues are fixed, and if time persists, attempt to add features which were discussed in the initial requirements, but are not on the list.	TBD

Table 1: Timeline

Task#	Description	Associated Milestone #	Start Date	Completion Date	Primary Responsibility
1	Create a basic application using Qt - Windows	1	Insert Start Date	Insert End Date	Deon
2	Create a basic application using PyQt - Windows	1			Deion
3	Create a basic application using	1			Wayne

	ImGui - Windows				
4	Create a basic application using GTK - Windows	1			James
5	Create a basic application using Qt - Linux	1			Deion
6	Create a basic application using PyQt - Linux	1			Deion
7	Create a basic application using ImGui - Linux	1			Wayne
8	Create a basic application using GTK - Linux	1			James
9	Create a debugger application that can attach itself to another program (Just a basic application, no breakpoints or anything)	1			Wayne
10	Implement a text viewer	2			James
11	Develop a set breakpoint system	2			Deion
12	Develop a breakpoint viewer system	2			James
13	Develop a visual indication of the program hitting a breakpoint	2			James
14	Implement a system to select an .exe to debug	2			Deion
15	Implement a system to select a running process to debug	2			Deion
16	Develop	2			Deion

	interactive buttons for debugger interactions				
17	Continue	2			Deion
18	Step forward	2			Deion
19	Stop	2			Deion
20	Implement: Read memory from an attached program -Windows	2			James
21	Implement: Write memory from an attached program -Windows	2			James
22	Implement: Read CPU registers from an attached program -Windows	2			James
23	Implement: Write CPU registers from an attached program -Windows	2			James
24	Implement: Set and manage breakpoints. -Windows	2			James
25	Implement: Start the debugee program from the debugger -Windows	2			James
26	Implement: Read in C++ source for user interaction -Windows	2			James
27	Create a debugger application that can attach itself to another program	2			Wayne
28	Implement: Read CPU registers from an attached program. -Linux	2			Wayne

29	Implement: Write CPU registers from an attached program. -Linux	2			Wayne
30	Implement: Set and manage breakpoints. -Linux	2			Wayne
31	Implement: Start the debugee program from the debugger. -Linux	2			Wayne
32	Implement: Read in C++ source for user interaction -Linux	2			Wayne
33	Implement a memory viewer/editor	3			Deion
34	Allow the user to view/edit the values in different bases	3			Deion
35	Develop an option to view assembly alongside source code.	3			Deion
36	Develop a syntax highlighting system (RegEx?)	3			Wayne
37	Implement for x86 Assembly	3			Wayne
38	Implement for C	2,3			Wayne
39	Implement for C++	2,3			James
40	Implement a command line interface to interact with the debugger	3			Wayne
41	Create a generalized platform layer where all operating system interactions have	3			Wayne

	to go through this layer. This will help make the debugger easy to port to new operating systems.				
42	Implement: Catch and interpret program exceptions. - Windows	3			James
43	Implement for other languages (add to this list)	4			Wayne
44	Implement some kind of communication system with a running debugger for remote controlling	4			Wayne
45	Test all code from Milestones 2, 3, and 4 and add additional features that were mentioned in requirements that would be great to have to enhance competitiveness (i.e. AI features, error interpretation, suggestions for runtime error repair/code changes), but are not necessary for a functioning debugger, may not be able to be completed by the project deadline, and has do be done at own risk without causing regressions in	5			Wayne, James, Deion

	prior functionality.				
# Table 3: Effort Matrix					
Task#	%Effort James	%Effort Wayne	%Effort Deion		
:-----:	:-----------:	:------------:	:-------------:		
1	2.5	95	2.5		
2	2.5	95	2.5		
3	2.5	95	2.5		
4	2.5	95	2.5		
5	2.5	95	2.5		
6	2.5	95	2.5		
7	2.5	95	2.5		
8	2.5	95	2.5		
9	2.5	95	2.5		
10	2.5	95	2.5		
11	2.5	95	2.5		
12	2.5	95	2.5		
13	2.5	95	2.5		
14	2.5	95	2.5		
15	2.5	95	2.5		
16	2.5	95	2.5		
17	2.5	95	2.5		
18	2.5	95	2.5		
19	2.5	95	2.5		
20	2.5	95	2.5		
21	2.5	95	2.5		
22	2.5	95	2.5		
23	2.5	95	2.5		

24	2.5	95	2.5		
25	2.5	95	2.5		
26	2.5	95	2.5		
27	2.5	95	2.5		
28	2.5	95	2.5		
29	2.5	95	2.5		
30	2.5	95	2.5		
31	2.5	95	2.5		
32	2.5	95	2.5		
33	2.5	95	2.5		
34	2.5	95	2.5		
35	2.5	95	2.5		
36	2.5	95	2.5		
37	2.5	95	2.5		
38	2.5	95	2.5		
39	2.5	95	2.5		
40	2.5	95	2.5		
41	2.5	95	2.5		
42	2.5	95	2.5		
43	2.5	95	2.5		
44	2.5	95	2.5		
45	2.5	95	2.5		