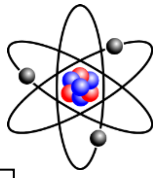
Name _____
Foundation of Programming

Miami Southridge High School
Bouwsma Rm 323



9007210: 16.04,13, 20.02-04, 21.02-04, 22.01,04, 23.01-04

Project 5 Biomechanics

We, as human beings walk on two legs. Biologists would tell you that, from an animal's viewpoint, humans are naturally acrobatic. Programmers, however, view human movement as simpler than animal movement, as we have already seen in our work up to this point. In this project, we are going to make a computer video game that uses ideas you got from the stick man, bomb and explosion examples in class. Feel free to use the internet; feel free to use the Explore text, but always respect copyrights.

Define your project (Grade1, by Inspection)

1. Choose a partner **you have never worked with before.** (Teams of two only, please.)
2. Decide on a game or video presentation topic with your partner.
3. Submit the name of your game or video presentation along with how to win or operate it to your instructor for a grade.

Design your Project

In this project, the design is yours. The process is yours. The subject matter is yours. We only ask that you keep your work suitable for any audience and that you not defame anybody or any institution.

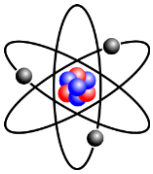
Your goal is to amuse. Your audience is the instructor, your parents and, eventually, the general public.

Eventually you and your partner will make this program into a phone app, so consider phone screen sizes as you design your program.[1] Your program needs to be attractive and usable on both a computer and a phone.

Brainstorm together about how to break down the work into parts that can be written as functions.

- You will need to **assemble the project** from the sections of code and functions that you both have produced. Inform the instructor detailing how you will accomplish assembly of the program.[2]
- Feel free to **draw** out how the program works. Show this to the instructor, as well.
- Work out all the **math** and get help where you need it. The instructor would be happy to help you with math ideas and principles. Feel free to use the internet, but be sure to keep a list of on-line sources and credit them in your program. Be sure the instructor knows who solved mathematical issues, to ensure accurate grading.

[1] Because graphics commands differ between different devices, professional programmers often put them in functions that can be modified to port their programs to new devices quickly.
[2] That makes partial credit grading possible in the event that your program never comes together.



- Be sure **your name** is on <u>all the work</u> you do by using # comments. At the top of the program **use comments** to list both of your names and indicate clearly which one of you assembled the program from the functions and code fragments you each wrote.

Suggestion1: Send each other functions and program fragments using email attachments, a flash drive or the cloud.

Suggestion2: The instructor is a shared resource. Ask questions and ask for **HOW TO** examples.

Code your project for a computer (Grade2, e-mail your program)

Use Thonny with Zelle's graphics package. Please remember to write small portions of code and test them before incorporating them into larger parts of the program. Test and debug everything. When you finish, email the project to your instructor for a grade.

Make it an App (Grade3, by Inspection)

Use a phone-based compiler with an IDE like **Pydroid 3** for android or **Pythonista** for iPhone.[3] Remember to include Zelle Graphics when you install to a phone.[4] You may use MIT's App Inventor 2, but you will need to learn how to use the canvas, and do a great deal of translation, if you choose to use their program.

Email your Python program to yourself and copy and paste it into the compiler. Get working and compile it for distribution.

Grading Rubric

You will be graded on scope, knowledge, robustness, and design.

1. **Scope** is the size and complexity of your program.
2. **Knowledge** is any, and all knowledge evident from your program.
3. **Robustness** is how much was debugged and how good is the user interface.
4. **Design** answers the question, is it fun and is it a good one of what it is.

	Category Name	Description	wt	Good	Bad	Challenge
1	Scope	size complexity thinking	3	large, complex, clever, many parts, subs, files, etc. detailed pictures, many options	repeated code, fluff, unexecuted code, short, trivial, bad or no graphics, few options, incomplete	instructor good students
2	Knowledge	code error free features	2	"best way" programming, many sensors used, cool techniques, optimized, advanced math, animation	clumsy code, easy to program style, inefficient, low performance, bad or no animation, incomplete	instructor
3	Robustitude	user interface error free exhaustive	1.5	No errors, intuitive controls, handles bad and weird cases, many issues solved for its scope	Syntax errors, "undocumented features," obvious program options or features missing, bad timing, incomplete	emotional people
4	Design	design conventional fun	1.5	Exhausts idea, interesting, easy to use, different, unique features, interesting options	disappointing, frustrating, few options, useless, unclear, ugly, incomplete	students

[3] You are not limited to the examples given above.

[4] Programmers call packages and modules that need to be included for their program to run dependencies. It is vital these be installed and be available to the program at execution time. It is vitally important to consider the demands all the dependencies make on the computer. If a small dependency requires high level computing power or demands unusual video characteristics, it will limit what economists call your customer base, or the people who could buy and use your program.