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 6 Clock

We are obsessed with time. Our schools have timed bells; our traffic is run on timed lights; our entertainment is runs across time slots. In section 1, you are going to retrieve the system time and use it to make a clock. Each level gives you points in the gradebook. In section 2, you will research management models used by software companies, and draw pictures of how they work for a grade. Then in section3, you will use what you learned to design, code, debug and deploy a clock app. There are three grades for this project.

1 Python Clock Program Learn the Math by Using It (Pairs)

Work **with a partner** in this section. (Groups of two only.) Divide up the work and email parts to each other. Programmers like to divide the work up into functions to make putting the project together easy. **YOUR NAME MUST BE ON ALL THE WORK YOU PRODUCE**, not your group names. Use # comments to label the sections of code you produce. **NO CREDIT WILL BE GIVEN TO CODE, IF THERE IS NOT A NAME ATTACHED TO IT.** **Keep this page for grading.**

Level 1: (1 pt)

Write a program to turn the entire screen into a large accurate digital clock. Your clock must be colorful and easy to read from across the room.

Here is a program that retrieves the date and the time to help you get started.

```
import time

while 1:
    t=time.localtime(time.time())
    print(t[3],":",t[4],":",t[5])

"""
  [0] -> year
  [1] -> month
  [2] -> day
  [3] -> hour
  [4] -> minute
  [5] -> second

"""
```

Level 2: (3pts)

Name ___________________
Foundation of Programming

Miami Southridge High School
Bouwsma Rm 323

Make a simple analog clock with hour, minute, and second hands that keep accurate time.

Here are some commands you may find useful:

`object.setWidth(pixels)` Sets the thickness of lines and circles.
`object.setFill(color)` Colors the inside of objects like circles and rectangles.
`setOutline(color)` Changes line and circle colors.

Hint: A full turn is 2π or 360°. A clock counts 60 tics as a full turn for minutes and seconds, and 12 as a full turn for hours.

Level 3: (1pt)

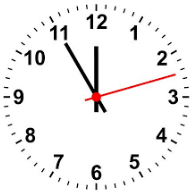
Put in hash marks for 12, 3, 6, and 9

Level 4: (2pts)

Change your clock so the hour hand moves small amounts as minutes elapse, so it can point between the numbers the way a real clock does. If your program works properly, the hour hand would be half way to the next hour when the minute hand reaches 30.

Level 5: (2pts)

Use math to fill in 60 small hash marks for every minute. The hash marks must always aim away from the center of the clock.



Level 6: (3pts)

Use trig to put in clock face numbers (1-12). They must be positioned correctly.

Extra Options = Extra Credit:

1. Put in the date. (1pt)
2. Put in the day of the week. (1pt)
3. Calculate the phases of the moon and display the moon. (6pts)
   <u>or</u> calculate the tides and display them.
4. Run an animation in the background. (2pts) with trig (5pts)
5. Your own idea (?pts)

Grading:

Any sufficiently advanced technology is indistinguishable from magic. – Arthur C. Clarke

Your grade will be calculated by adding half the number of points your group gets, rounded down, plus all the points you produce for your group. (A=12, B=9, C=6) Additionally, the first team to finish[1] gets 4 extra points – two points per person.

2 Software Development Models (Pairs)

Software companies try very hard to make high quality software for low cost and high profit. During the 1980s and 1990s the industry came into crisis. Hardware was being developed at such a fast speed that the software could not keep up. Software companies made changes to the way programmers create software.

1. **Research** some of these methods. Use some of these phrases to get started:
   - Software Development Cycle
   - Waterfall
   - Agile
2. **Draw** a picture of how each works. You can use the computer, or do it free hand. (You need good pictures of at least four models for an **A**)
3. List the **advantages and disadvantages** for each model in the picture or to the side of the picture. (At least three of each per picture are needed for an **A**)
4. **Choose** a development model for the next project. Explain why you chose the model in a sentence at the bottom of the page. Put your name on your work and turn it in for a grade.

Grading:

Your work will be graded by the following rubric:

1. Content and accuracy of pictures
2. Advantages and disadvantages are sufficient to decide between the models
3. Art quality & attractive design
4. Reasoning for decision

3 App Inventor 2 Clock Program Contest Your Design (Individual)

Use the MIT App Inventor 2 to create a clock app that will work on an Android Phone. The design is completely up to you. The way you code your clock is completely up to you. When you finish, install your app to the MIT AI2 Gallery under **RidgeClock+YourNumber**. You are allowed to help each other[2], but your work must be your own. You may research on-line, but your work must be your own. **Do not copy someone else’s code** and put your name on it.[3] Be honest.

Grading:

You will be graded in four areas by both your fellow students and by the instructor:

[1] By finished, it is assumed that at least one person has a B or better.

[2] Please remember that this is a contest. The people you help are your competitors. Kindness still dictates that you help those who ask.

[3] Programmers usually make money by writing programs. They view taking their code as stealing, and may sue, if they can prove they lost money because of your actions.

	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

Student grading instructions:

1	Worst in category
2	Very Bad
3	Bad
4	Low Average
5	Average
6	High Average
7	Good
8	Excellent
9	Best in category

Grade each app off the gallery with a partner who has a phone. Rate each app using a 1-9 scale, where 1 is the lowest and 9 is the highest. To make a good grade you must be careful to **average as close to 5 as possible** in each column.[4] You must give out **exactly one 1** in each column to the worst app in each category, and you must give out **exactly one 9** to the best app in each category. If you have graded correctly you will have exactly one 1 and one 9 in Scope, and one 1 and one 9 in Knowledge, and so on. When you finish evaluating the apps on paper, you will enter your evaluations into the grading computer, and the computer will grade your evaluation.

[4] If you don't average close to a 5, the grading program will change your numbers so they do average 5, and every number you enter will have fractions added and subtracted from it making every entry inaccurate. Computers are fair graders, but ruthless.