Why participate in this competition?

To prepare for the future that is already here, we're going to need more programmers! Robot ice cream makers? Burger chefs? Video game developers? Business owners? We all know that cars are becoming automated and if you can name a job with a repetitive task, it's likely there's an algorithm doing that job today too.

A 2015 Gallup poll sponsored by Google found that 9 out of 10 parents want schools to teach computer science - so our children will grow up not just *using* technology but learning how to *create* it. In fact, the majority of parents and teachers believe that computer science should be required for students to learn! A subsequent poll shows that 50% of parents consider computer programming the most important subject for students to learn after reading, writing, and math.

Codecraft partners with schools, universities and nonprofits to empower fun Computer Science education programs that improve the technical skills landscape and successfully narrow the STEM talent gap. Together we will improve students' confidence on the computer and their feeling of belonging in computing, therefore increasing the likelihood that they will further pursue computer science or engineering classes and successful careers.

Competition Overview

There are two competition categories within each 3 divisions by grade level; elementary, middle, and high school. In each category and division, awards for 1st, 2nd, and 3rd place, depending on the category, will be given. Each student competitor may choose, with guidance from their teacher, coach, or mentor, the competition category they wish to compete in.

Division	Category 1	Category 2
Elementary	Scratch Programming "Game"	Scratch Programming "Storytelling"
Middle School	Scratch Programming "Game" or "Storytelling"	Python "Tic-Tac-Toe"
High School	Python "Tic-Tac-Toe"	Java "Tic-Tac-Toe"

Elementary division projects in either category will be made using the Scratch computer programming platform and judged on its engagement, artwork, use of digital media, use of computer science concepts, originality, and completeness (see Scratch rubric below).
NOTE: A maximum of 10 project submissions per elementary school will be accepted.

Middle School Category 1 projects will be made using the Scratch computer programming platform (see Scratch rubric below), and Category 2 projects will be written using the Python programming language (see Tic-Tac-Toe instructions below).

High School Category 1 projects will be written using the Python programming language, and Category 2 projects will be written using the Java programming language (see Tic-Tac-Toe instructions below).

2019 - 2020 General Competition Rules

1. The competition is open to elementary, middle, and high school students.

2. Elementary division competition project must be created using the Scratch programming platform (scratch.mit.edu). Middle school division can choose to enter Scratch or Python projects (python.org). High school division can enter Python or Java projects (java.com).

3. To be considered, a project submission **MUST** be submitted **NO LATER** than midnight on January 17, 2020, here: code.cr/SECME2020

4. The competition organizers reserve the right to disqualify any entry based on inappropriate or copyrighted content and any entries which do not adhere to the competition rules and guidelines.

5. When an entry is submitted, permission is granted to the organizers of the competition to make unrestricted use of the entry in the future for publicity or educational purposes. In such use, the organizers will make sure that the author/school is clearly acknowledged, with consent documented and privacy in mind.

Competition Project Requirements

1. Projects must be original works by a student creator or team (up to 2 students).

2. Entries **must be ORIGINAL works created by the team or individual** submitting the entry.

3. If an entry incorporates music, sound, text or images, you must own the rights to use that material, or provide creative commons attribution in the project “Notes & Credits” Section.
4. No violence or simulation of violence. Use your programming powers for good or positive change!
5. Project content is limited only by your imagination, ability to plan and demonstration of your programming ability.
6. Scratch projects must have clear, precise and appropriate Title, Instructions, and Notes or Credits.

Prizes and Awards

The following will be awarded for each division; elementary, middle, and high school:

Division	Category 1	Category 2
Elementary	1st, 2nd, and 3rd place Scratch “Game”	1st, 2nd, and 3rd place Scratch “Storytelling”
Middle School	1st, 2nd, and 3rd place Scratch “Game” or “Storytelling”	1st and 2nd place Python “Tic-Tac-Toe”
High School	1st and 2nd place Python “Tic-Tac-Toe”	1st and 2nd place Java “Tic-Tac-Toe”

Tic-Tac-Toe

Judging

- Players’ programs will be randomly matched against each other in a bracketed tournament. There will be multiple games of tic-tac-toe per matchup.
- The victor of a matchup will be whoever wins the most tic-tac-toe games. In the event of a draw, the “player” that ran most efficiently (i.e. least amount of time to execute) will be the winner. Winners will advance through the bracket until there is a single “player” left standing.
- “Player” matches will be run through our interface. (Details below)

- If a “player” submits an invalid move, their turn will be skipped. If a single game lasts more than a certain amount of turns (in excess of 9), the “player” that ran most efficiently will be the winner.
- If a “player” fails to submit a coordinate after their turn is over, crashes, gets stuck in an infinite loop, or fails to compile, they will be disqualified.

Interface Details & Requirements

- Competitors’ classes must inherit from class “Player” and implement the following:
    - **calcMove()**
        - **Abstract** and **void** function, to be implemented by students. This is the function that is called by the judging program on each player’s turn.
    - **Java : submitMove(int row, int col) , Python: submitMove(row, column)**
        - Function for submitting a move. If the move is invalid (e.g. array (or list) coordinates are out of bounds or a tile already occupied by an ‘X’ or ‘O’) the move will simply be ignored and move on to the next players turn. **The submitMove() function must be called from within calcMove().**
    - **getGameState()**
        - Function that returns two-dimensional array (list in Python) of chars on the tic tac toe board; empty tiles will be represented as a space char ‘ ‘, X by ‘X’ char and O by ‘O’ char.
        - The coordinates of the Board are arranged in the following way:

[0] [0]	[0] [1]	[0] [2]
[1] [0]	[1] [1]	[1] [2]
[2] [0]	[2] [1]	[2] [2]

        So for example the index [0][2] of the array (or list) would return the value of the upper right-most square

        - NOTE: **X and O will returned as CAPITAL letters**. Team “X” will always go first.
    - **getTeam()**
        - function that returns an ‘X’ char or ‘O’ char depending on which team the game has selected for them. Use this to determine whether the player is playing as X or O.

- NOTE: **X and O will returned as CAPITAL letters**. Team "X" will always go first.

Program Specifications

Python:

- Use Python version 3 (3.7.4 recommended)
- The submitted file should be named in the format **lastname_FirstName.py**
- The code must define a function calcMove() that will be called. All logic should be contained inside this function. (You may define your own auxiliary functions outside of calcMove(), but they must be called inside of calcMove() to be used)
- Comment your name(s) at the beginning of the file

Java:

- Use Java version 12
- Filename should be **lastname_firstname**(replace with student's name)
- Code should be contained in a class called **lastname_firstname** (replace with student's name) and inherit from (extend) the class **Player** (see template below)
- Comment your name(s) at the beginning of the file

Example Java Template:

```
public class Smith_John extends Player{

        calcMove(){
                //Code here to determine the coordinates of the players move, use
                //getGameState() to determine the layout of the board
                submitMove(Xcoordinate, Ycoordinate);
        }
}
```

Example Python Template

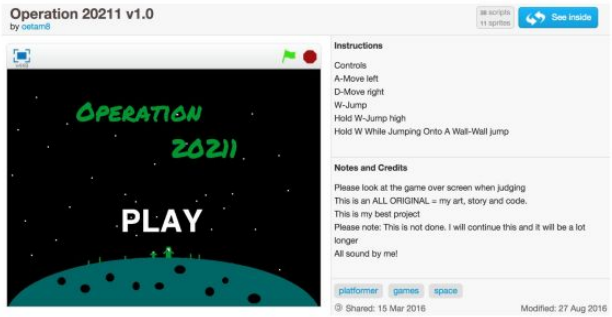
```
#imports can go here if necessary
def calcMove():

        #Code here to determine the coordinates of the players move, use
        #getGameState() to determine the layout of the board

        submitMove(Xcoordinate, Ycoordinate)
```

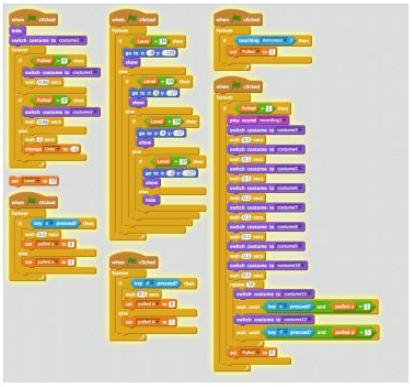
Scratch Programming

Judging

1. Each entry will be reviewed by a panel of at least 2 judges and scored for its engagement, artwork, use of digital media, use of computer science concepts, originality, and completeness . The score sheets below will be used by the Judges during the competition.

2. The decisions of the judging panel are final and no correspondence will be entered into. The scorecard sample below will be online as a digital form and made available to each judge for use as they review the projects.





Scratch Rubric				
				Points
Engagement				
	1 2 3	4 5 6	7 8 9 10	
Artwork				
	1 2 3	4 5 6	7 8 9 10	
Digital Media				
	1 2 3	4 5 6	7 8 9 10	
Coding / CS Development				

CODECRAFT WORKS

	1 2 3	4 5 6	7 8 9 10	
Originality				
	1 2 3	4 5 6	7 8 9 10	
Completeness (Testing / QA)				
	1 2 3	4 5 6	7 8 9 10	
Totals				