RECRUITMENT SCREENING TEST
SOFTWARE MODULE

INSTRUCTIONS

· This screening test consists of two sections, A and B.
· Section A tests you on ROS & Section B tests you on specific aspects.
· Questions in Section A are compulsory. Try to attempt as many Questions as possible in Section B (Each Question in this section has a weightage ranging between 0-1). Don't worry about the scoring. Please do note that even if you can't get a question right,do mention its approach.
· Submission format: A github repo containing all the required files and a Readme.
· The codes must be well written and understandable .
· Use PEP8 styling for Python and follow Google Styling Guide(GSG) for C++ .
· Any kind of plagiarism will be dealt with extremely. Acknowledge any and every resource used.

Note: Submissions must be proper code in a github repo.

The README should be in the following format

Submission for Team Abhiyaan
==========
(Example)

Name:
Tony Stark

Roll Number:
XX01S001

Previous Experience:

Dead in End Game :(.
I made an Unmanned vehicle that detects corona virus and kills it!!
I made Jarvis.

Current PORs:

I ammmmm Iron Man!!
Member of The Avengers!

Why I want to work in the team:

Because I think the team is the best in insti ;)

Relevant Courses:

In Institute

CS1100
ME2200
CS6251

Passed
W - Attendance Shortage
Doing

Online

CS20SI
Andrew NG Deeplearning.ai
David Silver RL

· Mail the github repo link to abhiyaan@smail.iitm.ac.in with your name and roll number.
· The subject of the mail should be "**Abhiyaan || Software Submission || <your name>**" .
· **All the very Best. Hope to see you in our team soon!!**

SECTION - A:

This section is compulsory. This section tests you on general topics like Data structures and Algorithms. You will also be tested on your ability to write code to work with the ROS framework.

ROBOT OPERATING SYSTEM (ROS)

Welcome to the World of ROS!!
If you are an absolute beginner in ROS, you might want to checkout : ROS Tutorials . You can try '**The Construct**' channel on YouTube as well.

First Step: Install ROS Kinetic in Ubuntu 16.04. You can use a live USB disk or install Linux as dual boot.

1. Create a catkin workspace after installing **ROS kinetic**. Do the following tasks inside the workspace.

2. Write a ROS node in C++/Python to publish the string "Welcome to Abhiyaan" to the
Topic:
/welcome_message
Dtype:
std::string

3. Write a ROS node in C++/Python that listens to the
Topic:
/welcome_message
Dtype:
std::string
and prints the content of the message on the terminal.

Before going to the next question, you might have to read about turtlesim in ROS. Chins up fellas!! You are going to encounter a virtual robot!!

4.Open turtlesim.
  Spawn a turtle name '**abhiyaan**' at (10,10) at 0 degrees using the command
  rosservice call /spawn "x: 10.0
  y: 10.0
  theta: 0.0
  name: 'abhiyaan'"

The command rosservice comes under a separate feature of ROS called services & clients. You need not worry about it now ;).

  Write a node named '**turtle_exercise**' such that it subscribes to relevant topics and publishes to relevant topics to make 'turtle1' move towards 'abhiyaan' and stops just before the distance from abhiyaan is 2 units.

Hints
*** rostopic list** *** rosmsg show**
*** rostopic info**

You might have to use the above commands as a weapon to solve this.

SECTION - B:

This section evaluates your interest in the sub topics. We suggest you try out everything! Even if you do not know anything, you will really enjoy learning and doing it!

SUBSECTION B.1 :DATA STRUCTURES

1. Make an arithmetic expression evaluator using stack data structure. (**Weightage: 0.6**)

2. Create three classes in C++

   a. Master
   b. Publisher
   c. Subscriber

Master class contains the list of subscriber classes & Publisher Classes and the topics corresponding to it. Publisher class publishes a string in a topic to Master class. Now Master sends the string to all nodes subscribed to that topic.

After this subscriber saves the given string. (**Weightage: 0.6**)

Note: Also State the assumptions made in both the questions along with your code.

SUBSECTION B.2 : CONTROLLER

1. This is a theoretical Question but is indeed an interesting problem generally encountered when developing a navigational robot!! Consider our Virat which has planned its path to reach a destination. The bot starts to move. How will it know that it is following the path it has planned. Also the path planned will be changing continuously in case there are obstacles coming in between. This is where a controller comes into play!! There are many ways to make a good controller. Say you are the controller expert of the team. Suggest us a way in how you will approach the above problem. (**Weightage: 0.8**)

SUBSECTION B.3 : COMPUTER VISION

OpenCV is the most used image processing library in the planet! So it is a crime to not include any questions from the same topic!

1. One of the important tasks that an autonomous car must do is to identify **Lanes** and follow proper protocol.
Task: Say you have images of lanes such as this, where the lanes are straight lines. Use methods of Computer Vision to return the image with the lanes highlighted.
Note: Use methods such as Hough Transform to achieve this.

Example:
Input → Output
(**Weightage: 0.7**)

2. **Traffic Light Detection** is a major task that any autonomous car must do.

   a. **Task 1:**
      Given a set of images such as this one, write a python script (using OpenCV library - not necessary) to return what signal it is showing.

   b. **Task 2:**
      Let's take this one step further. Say you're given an image of a road junction such as this (what may be taken from our self-driving car), find out a way to isolate the traffic lights from these images(draw a bounding box).
      In case you need a training dataset, use the following one: dataset

(**Weightage: 0.7**)

SUBSECTION B.4 : COMMUNICATION AND NETWORKING

1. Use Jaus++ Library (http://active-ist.sourceforge.net/jaus++.php?menu=jaus) and setup a JAUS component with the following core services
Discovery
Transport
Liveliness

Print a message when a Query is received.

Your submission must include another component that sends the following queries to the
main component.
QueryIdentification

QueryTransportPolicy

QueryHeartbeatPulse
It should also print Reports received after queries.

(**Weightage: 0.6**)