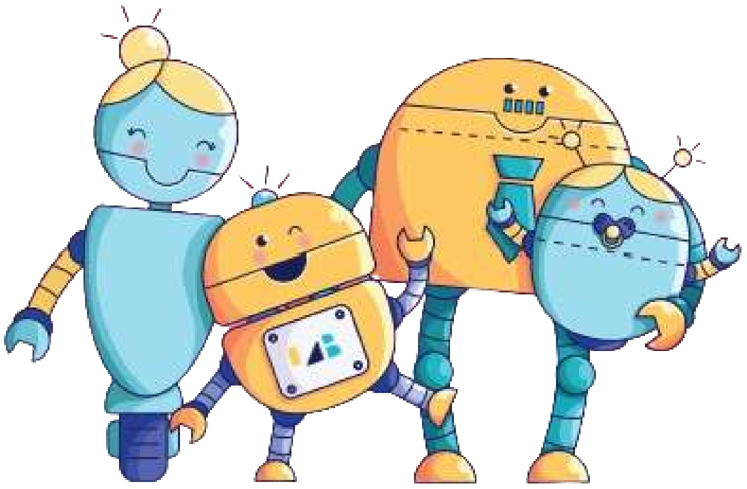
THE LAB

LEARNING WITHOUT BOUNDARIES

THE LAB CODER

Childhood and early adolescence are the critical age ranges for children to learn anything, including Coding, because their brains are still developing and learning "how to learn".

Now is the chance to introduce your child to native programming.



MEET THE SENIOR TEAM

DR. OKA KURNIAWAN

Dr. Oka is a Senior Lecturer for Singapore University of Technology and Design. His research areas include Computer Science Education.

CURRICULUM SPECIALIST

DR. SCARLETT MATTOLI

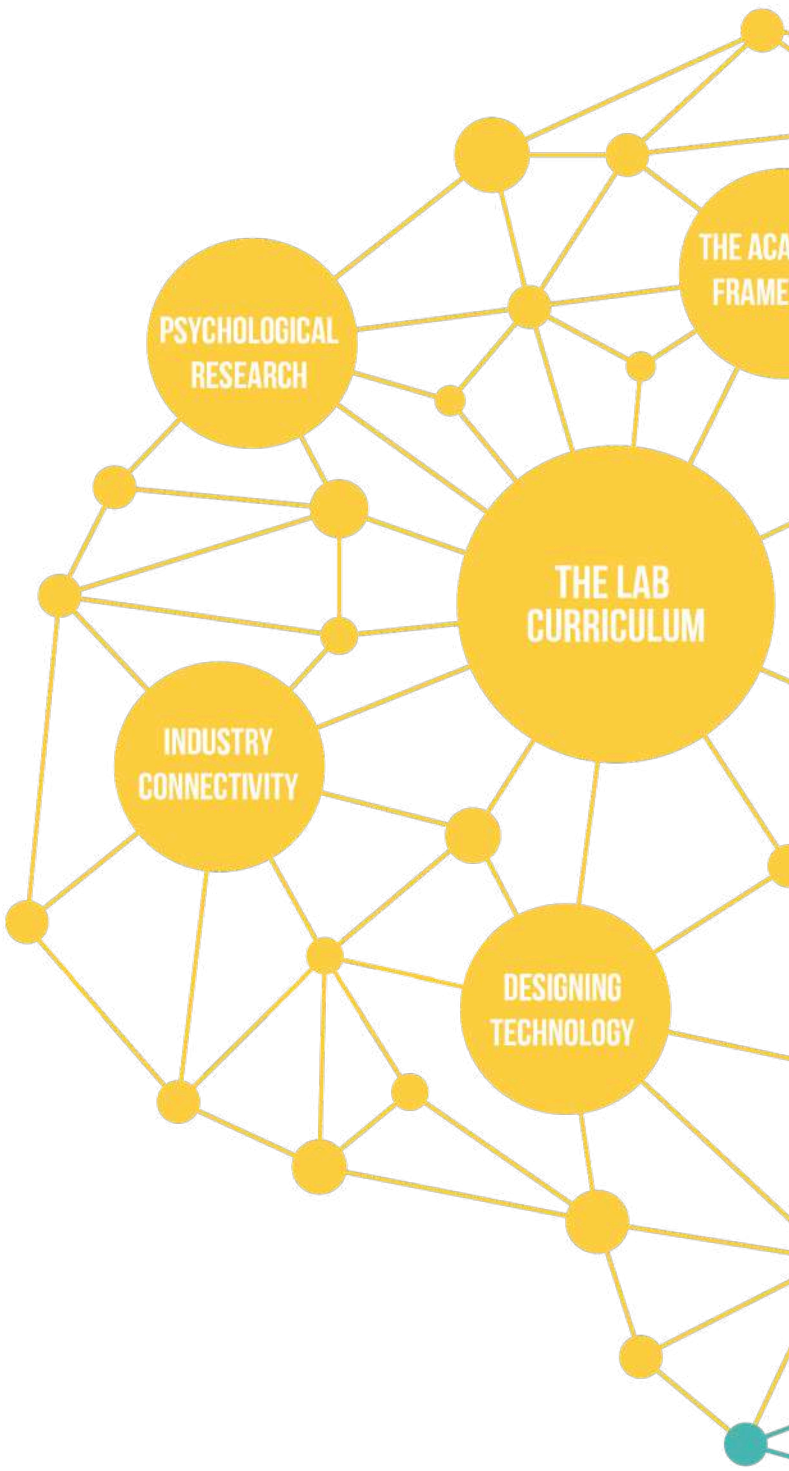
Dr. Scarlett is a Psychotherapist/Counsellor, Coaching Psychologist & Supervisor and Psychometrist, specialising in psychological and therapeutic support.

CHILD PSYCHOLOGIST SPECIALIST

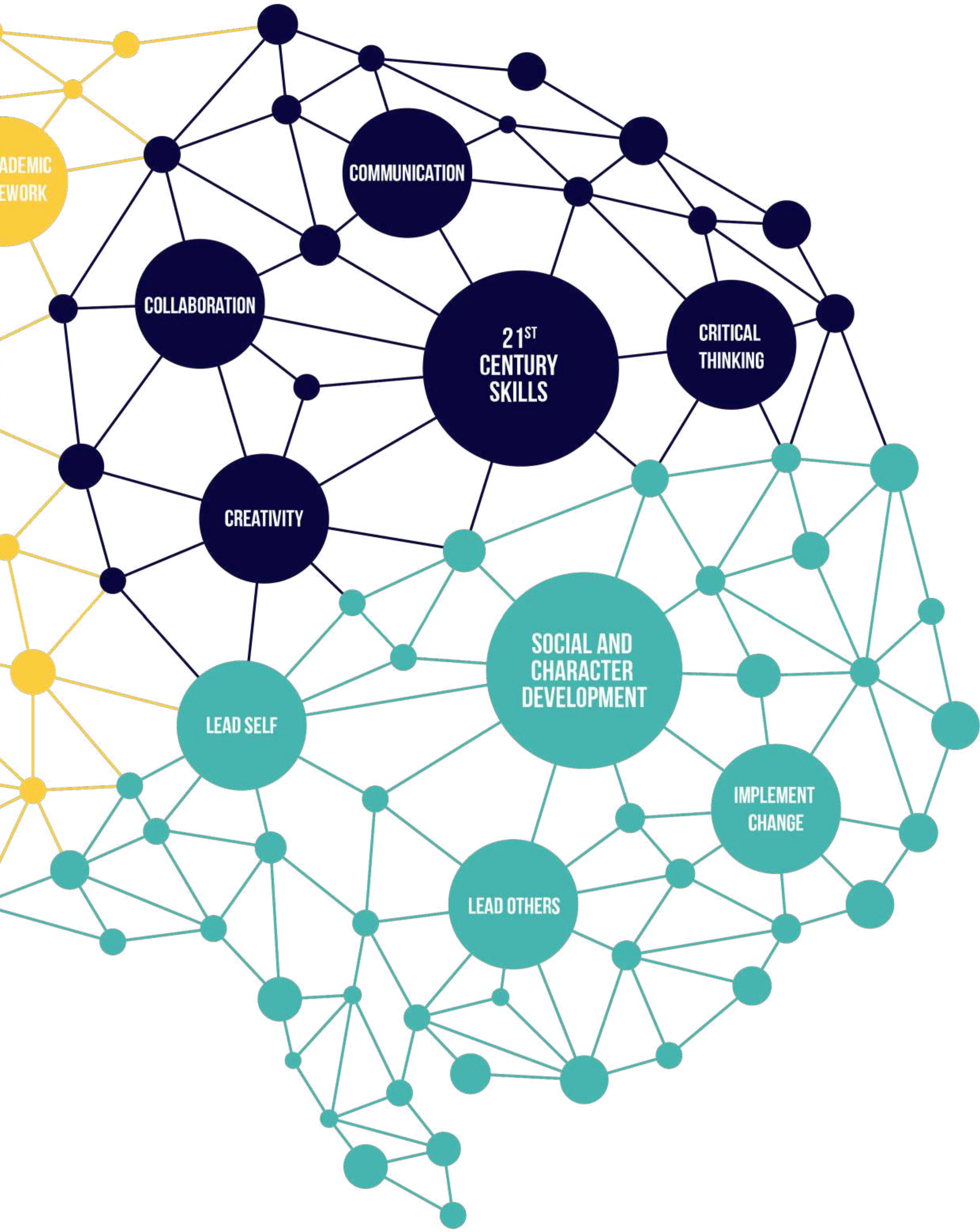
DR. COLLIN ANG

Dr. Collin is the Managing Director of Decision Science and is a thought leader in the industry for digital transformation and analytics

TECHNOLOGY SPECIALIST


PSYCHOLOGICAL RESEARCH
THE ACA
FRAME
THE LAB CURRICULUM
INDUSTRY CONNECTIVITY
DESIGNING TECHNOLOGY

CURRICULUM


ADEMIC
EWORK
COMMUNICATION
COLLABORATION
21ST CENTURY SKILLS
CRITICAL THINKING
CREATIVITY
SOCIAL AND CHARACTER DEVELOPMENT
LEAD SELF
IMPLEMENT CHANGE
LEAD OTHERS

EMPOWERING STUDENTS THROUGH COMPUTATIONAL THINKING

THE LAB
LEARNING WITHOUT BOUNDARIES

FOUNDATION

FOR AGE 10 & 14 YEARS OLD

At the end of this course, your child will achieve a high competency in Computational Thinking and code proficiently in the Python programming language.

This curriculum is reviewed by Dr. Oka Kurniawan, Senior Lecturer SUTD

Program Outline

- Open lab structure
- Student-centered, inquiry-based curriculum
- 4 Levels (Foundation,
- Basic, Intermediate, Advanced)
- Fuses Coding with multiple disciplines
- Ratio 1:8
- Duration 100 mins

JOIN US FOR A FUN-FILLED LEARNING EXPERIENCE!

FOUNDATION

The Lab Coder Foundation is a stepping stone to the Lab Coder program. It serves as a preparatory program for students to ease them into the vigorous requirements of the Lab Coder program. It provides a broad introductory to allow students to seek the skills of a good programmer.

The curriculum focuses mainly on developing the 4 core skills to prepare the student for The Lab Coder program.There are (1) Observation: (2) Analysis: (3) Visualization and (4) Debugging.

LEVELS	**2**			
PROGRAMMING CONCEPTS	Observation Students will be given several pre-coded programs and are tasked to decipher what the codes mean in their assigned challenges. These exercises are aimed to help students in their observation skills.	Analysis Students are given a series of challenges that focuses on improving their analytical thinking and logical thinking skills.This will help them to solve future coding challenges in a more logical and analytical manner.	Visualization Students are given a series of challenges aimed at improving their visualization skills.The purpose is to train students to easily identify trends. patterns, and outliers within large information data.	Debugging The debugging process usually consists of the following: examine the error symptoms, identify the cause, and finally fix the errors. Students are trained with a series of challenges that are purposely coded erroneously and through these exercises, learn to cultivate strong debugging skills

BASIC

This curricutum is a fun and interactive introductory course to students with tittle or no prior experience in Python. It is designed for beginner's level introduction to visual programming, Python, and robotics. In this course, students will learn how to build their own mini projects revolving a Raspberry Pi, understand its components and execute commands through basic visual programming.

Core computational thinking concepts such as decomposition, pattern recognition, and abstraction will be introduced as will programming toots such as flowcharts. The curriculum covers Python programming concepts, including sequencing, programming loops, conditional statements, ond operators.

LEVELS	1				
PROGRAM MING CONCEPTS	Basic Loop A For Loop is used for iterating over a sequence.This is one of the most basic concepts and is also highly used in programming.		Conditional Statements The basis of logic is contributed largely by if-else statementsCoupled with AND/OR operators. multiple conditions can be constructed to form complex decision-making processes.Examples: If, If/Else, If/Else/If		Operators Understanding the use of operators. not just for arithmetic operations but for other data types as well to construct appropriate conditions for conditional checksExamples: >, <, =
ROBOTIC SENSORS	LCD - Set Display Type - On - On for seconds - Set column and row - Clear Screen	Buzzer - Set play type - On - On for seconds - Play music note - Play tone frequency	Button - Wait for button pressed - Wait for button released - Wait for button Bumped - Return button type	Ultrasonic Sensor - Distance in CM	Colour Sensor - Return Colour Value - Return Colour Name - Return Ambient Light

THE LAB coder

INTERMEDIATE

During the course students will take the concepts that they have learnt in Basic to the next level. More advanced Python programming concepts will be introduced to the students to ensure they have programming thinking capabilities similar te a university undergraduate. The curriculum covers Python programming concepts, including more complex programming loops, nested conditional statements, variables and lists.				
LEVELS	**2**			
PROGRAMMING CONCEPTS	While Loop This is an extension of For Loop. While Loop, a condition triggered loop that allows you to formulate cycles without the need to know the definite times of repetition.	Nested Loop (For or while) A nested loop is a loop inside a loop. The 'inner loop' will be executed one time for each iteration of the 'outer loop'.	Variables Variables play an important role in computer programming because they enable programmers to write flexible programs. Rather than entering data directly into a program. a programmer can use variables to represent the data.	List Extending the programming functionality beyond basic applications with the use of list to handle large or scalable data storing. Powerful constructs can be formed with loops to solve complex problems with short codes.

INTERMEDIATE

PROGRAMMING CONCEPTS	Function Breaking codes down into functions are the norm. Not just for readability but also, for programme optimisation, ease of debugging and even feasibility of a solution. Particularly, functions with input parameters and return values are usually the indispensable assets of a programme.	Conditional Statements The basis of logic is contributed largely by if else statements. Coupled with AND/OR operators, multiple conditions can be constructed to form complex decision making processes. Examples: If, If/Else,If/Else/If,Multiple If/Else statements	Operators Understanding the use of operators, not just for arithmetic operations but for other data types as well to construct appropriate conditions for conditional checks. Examples: >. <, = =	List Extending the programming functionality beyond basic applications with the use of list to handle large or scalable data storing. Powerful constructs can be formed with loops to solve complex problems with short codes.
ROBOTIC SENSORS	LCD - Set Display Type - On - On for seconds - Set column and row - Clear Screen	Buzzer - Set play type - On - On for seconds - Play music note - Play tone frequency	Button - Wait for button pressed - Wait for button released - Wait for button bumped - Return button type	Ultrasonic Sensor - Distance in CM
	Colour Sensor - Return Colour Value - Return Colour Name - Return Ambient Light	Motor driver - Set motor type - On - On for seconds - Turn clockwise/anticlockwise - Max/min speed	Gyro Sensor - Return X Y Z axis - Reset X, Y, Z axis	

THE LAB coder

ADVANCED

Upon strengthening their computational thinking in our The Lab Basic and intermediate curriculums, students will progress into code implementation. The advanced curriculum will train the students on Python language syntaxes of various programming concepts, including those they have encountered during the basic and intermediate curriculums. *In order to expose the students to a vast range of real-life problems, the advanced curricutum focuses on algorithmic development. Practical and interesting challenges from different domains are carefully curated and customised for progressive training. The completion of this course enables them to have an in-depth knowledge of modern-day programming, as well as the understanding of the level of versatility required for a programmer's skills to be useful.*			
LEVELS	**3**		
PYTHON PROGRAMMING TOPICS/CONCEPTS	Screen Input/Output Use of different print and input formats to control the display of information on the screen and capturing of data entries from the user.	Function Breaking codes down into functions is the norm. Not just for readability but also for programme optimisation, ease ofdebugging and even feasibility of a solution.Particularly, functions with input parameters and return values are usually the indispensable assets of a programme.	OOP Object-oriented programming(OOP) Is the modernprogramming methodology compared to procedural programming. Learn about how this methodology changes the way a solution is implemented with the same computational thinking.

ADVANCED

PYTHON PROGRAM MING TOPICS/ CONCEPTS	Variables, Data Type and Casting Extending from the knowledge of a variable, learn about what data type of a variable means and how to convert between the different types for appropriate operations.	2D List A list can go multi-dimensional. By just adding a second dimension, 2D list gives a new perspective on how problems can be effectively represented and their solutions becoming more obvious.	OOP - Python Class The basis of OOP is what we call a class. Learn how to build classes and create 'objects' from these classes to execute your codes (thus the term object-oriented programming).
	Operators Understanding the use of operators, not just for arithmetic operations but for other data types as well in order to manipulate the data or construct appropriate conditions for comparisons.	Dictionary A dictionary is a collection of key-value pairs which allows each value to be instantly accessed by providing its key. This data structure stands out in applications where you need to regularly search for data with a unique key.	OOP - Class/Object variables Understanding the difference between class and object variables helps you to design your classes with variables that can be shared by its objects.
	For Loop More than just a repeat cycle. Learn when to deploy the for loop and how to use the counter in the loop as part of your algorithm.	Turtle Extending beyond text-based display, the graphic library, Turtle, provides a means to illustrate on the display with colourful lines and curves. Graphics are not just a good-to have, but a pre-requisite in some applications such as games.	OOP - Static methods Creating functions in a class that can be called without object instances, called static methods, is one of the variants to designing functions in OOP.

THE LAB coder

ADVANCED

PYTHON PROGRAM MING TOPICS/ CONCEPTS	While Loop Condition-triggered loop that allows you to formulate cycles without the need to know the definite times of repetition.	List Extending the programming functionality beyond basic applications with the use of list to handle large or scalable data.Powerful constructs can be formed with loops to solve complex problems with short codes.	OOP - Inheritance Inheritance allows us to define a class that inherits all the methods and properties from another class. This is useful for code extension without re-implementation. You'll be accustomed to terms like 'Parent Class' and 'Child Class'.
	Conditional Statements The basis of logic is contributed largely by if-else statements. Coupled with AND/OR operators, multiple conditions can be constructed to form complex decision-making processes.	Nested Loops/Conditional Statements Nesting will be commonly used as the problems increase in complexity.Nesting involves nested loops as well as nested conditional statements.	OOP - Polymorphism Polymorphism means the ability to take various forms. In Python, Polymorphism allows us to redefine functions existing in an inherited class, thereby changing its functionality to suit the inheriting class.
	Built - In Functions Along the way, you will be introduced useful built-in functions such as random, sleep, split, etc, which will be become useful tools for your algorithms..	String Manipulation Many problems boil down to solving string patterns. Hence, efficient ways to manipulate strings are vital in formulating solutions to such problems.	File I/O A programme will usually need to save data into the harddisk for subsequent retrieval. The knowledge of File I/O is, thus, essential for understanding how database works.

THE LAB coder

JOIN US AT

COMMIT TO A YEARLY MEMBERSHIP
&
GET PROMOTIONAL RATES!

Enjoy up to two visits per week with your membership

3 months	$380/mth
6 months	$340/mth
12 months	$320/mth

*** Registration fee is $80 per student*

SIGN UP NOW AT
WWW.THELAB.SG/SIGNUP

CONTACT US

North East: Buangkok Square
Blk 991 Buangkok Square #03-07
Singapore 530991

North: Woods Square
12 Woods Square Tower 1 #04-68
Singapore 737715

East: Kinex Mall
11 Tanjong Katong Road
#03-01/02 Singapore 437157

Central: Taste Orchard
160 Orchard Road #03-07
Singapore 238842

contact@thelab.sg

(+65) 8916 0017

THE LAB
LEARNING WITHOUT BOUNDARIES