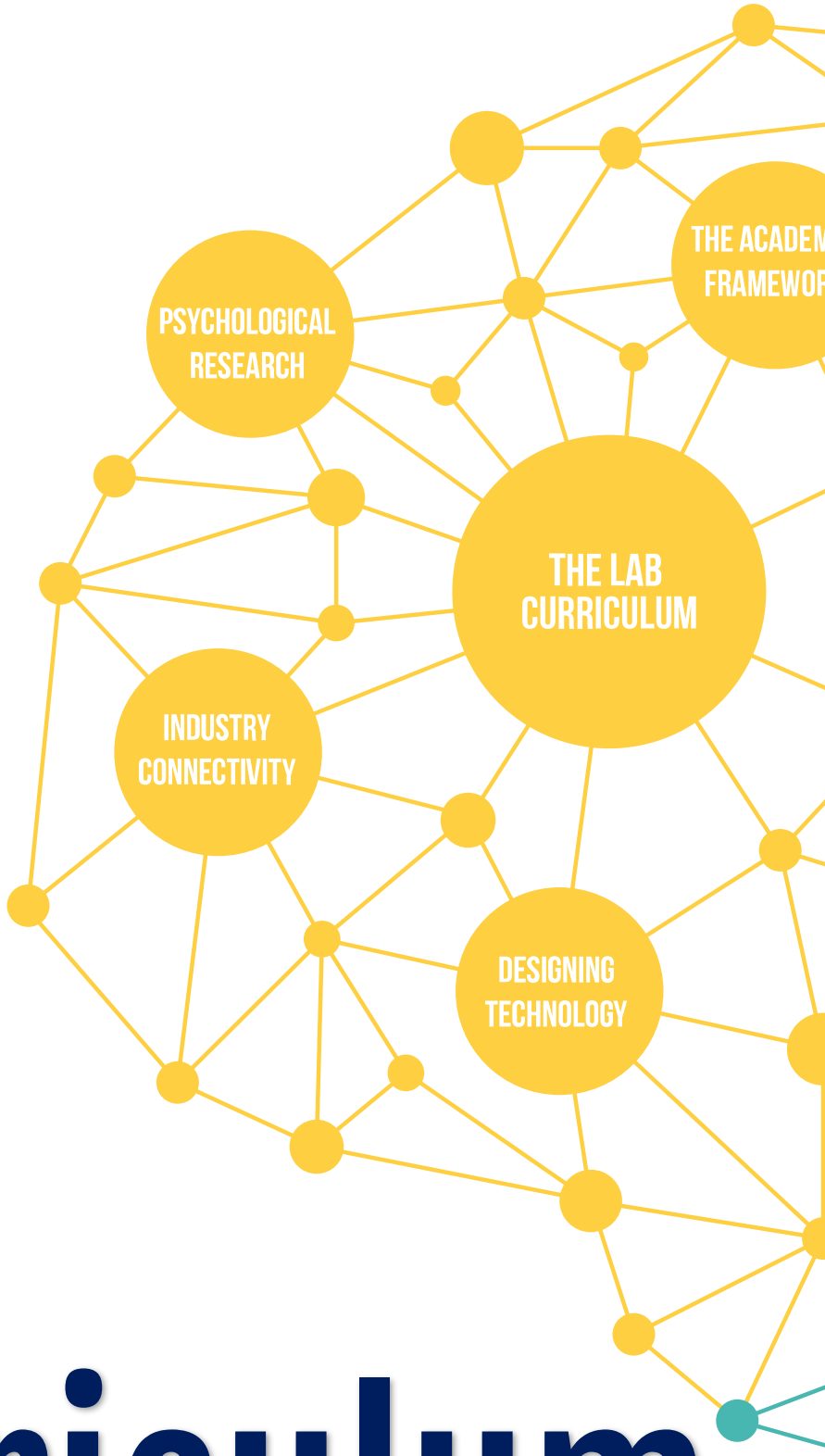

THE LAB
LEARNING WITHOUT BOUNDARIES

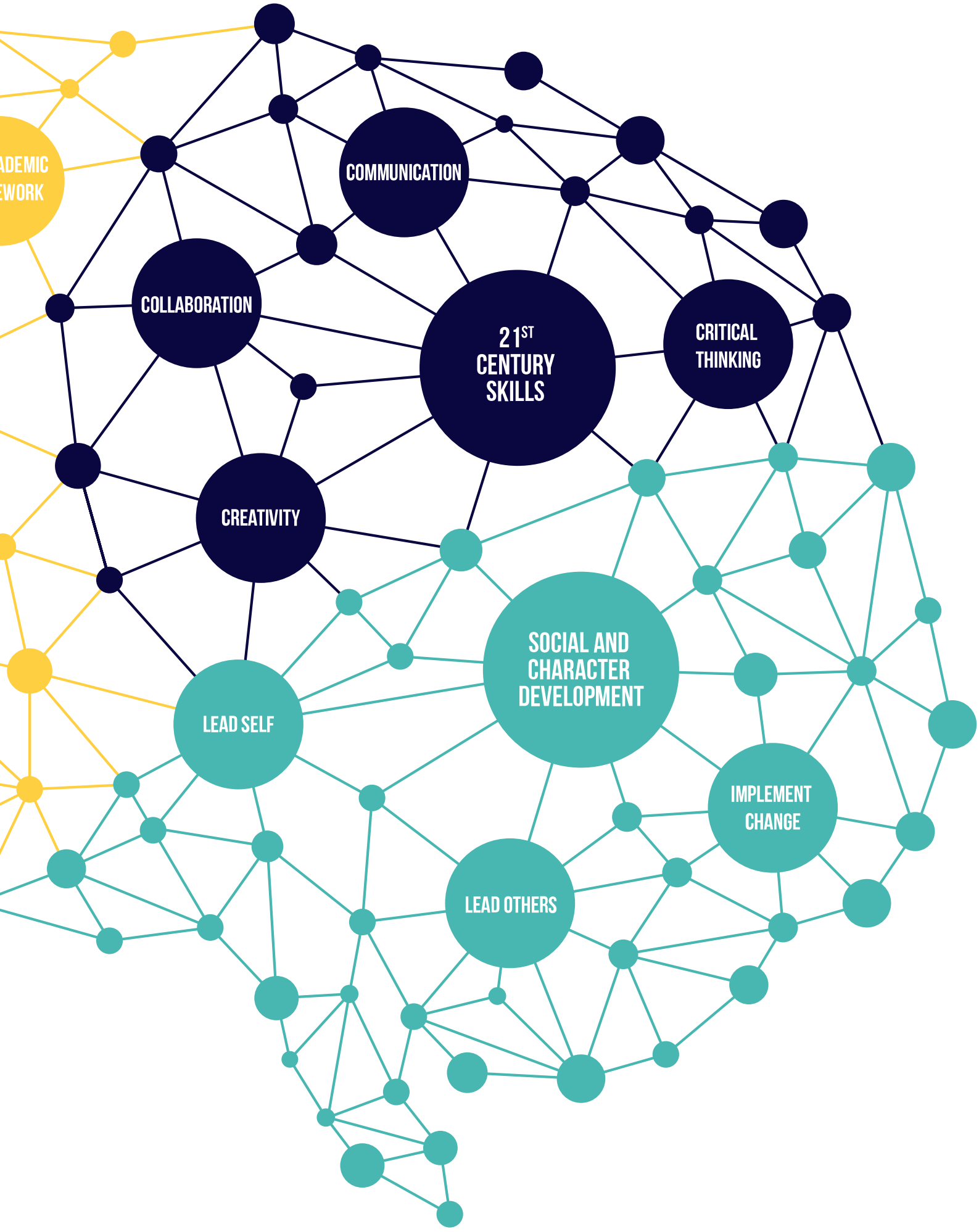
THE LAB X

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.


PSYCHOLOGICAL RESEARCH
THE ACADEM
FRAMEWOR
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

Senior Team

Dr. Oka Kurniawan
The Lab Curriculum Specialist

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

Dr. Scarlett Mattoli
Child Psychologist Specialist

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

Dr. Collin Ang
Technology/Industry Specialist

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

Students

Empowering through Computational Thinking

THE LAB
LEARNING WITHOUT BOUNDARIES

The Lab X Program

The Lab X is a series of programs suited for students with varied interests. These programs cater to students who have completed The Lab Coder Advanced course and are interested to expand their coding knowledge into different specializations.

The Lab Python Programming

The program is for young adults who are new in programming. Python is one of the simplest open-source programming languages to learn, making it a great entry point for beginners interested in data science.

The Lab Unity Game Development

For students who are interesting in games development, The Lab Unity Game Development will be an exciting continuation of the students' interest into the world of coding and technology.

The Lab Competitive Programming

The Lab Competitive Programming is a training course for advanced students to showcase their programming prowess through participation in international competitions.

PROGRAM OUTLINE

THE LAB X - PYTHON PROGRAMMING
(The Lab Coder Advanced Course)

The advanced curriculum trains the students on Python language syntaxes of various programming concepts for practical business use.

In order to expose the students to a vast range of real-life problems, the advanced curriculum 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 of debugging 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 modern programming methodology compared to procedural programming. Learn about how this methodology changes the way a solution is implemented with the same computational thinking.

PYTHON PROGRAMMING 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.

PYTHON PROGRAMMING 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.

PROGRAM OUTLINE

THE LAB COMPETITIVE PROGRAMMING

The Lab Competitive Programming are catered to individuals who have a strong foundation in Mathematics, a passion for programming and the fortitude to persevere through countless hours of thinking through highly difficult coding challenges. Our selection process requires all interested students to take an entry examination.

Competitions play a role in motivating students to perform and excel and offer a lot more reward than just the winning prize. They offer a chance for participants to gain substantial experience, showcase skills, analyze and evaluate outcomes and uncover personal aptitude.

TRAINING TIMELINE		CONTENT FOR EACH LESSON
Dates	**Description**	
Feb till End of March	Application	
1st week of April	Application closes	
1st and 2nd week of April	Entry Level Exam	
3rd week of April	Marking Period	
4th week of April	Announcement of results	
1st week of May	Commencement of training Content for each lesson: - Lecture (topics to be spread across training period) – Practice on competition questions and solution walkthroughs - Homework may be given in some lessons	
Feb of next year	CCC Competition	
May of next year	Code Quest Competition (* Only for student above the age of 14 in May)	

- Teaching of theories including useful mathematical functions, programming paradigms, data structures, computational geometry, popular algorithms, etc.
    - Practice on competition questions and solution walkthroughs
- Homework may be given in some lessons

PROGRAM OUTLINE

THE LAB UNITY GAME DEVELOPMENT

Unity is the most popular game engine in the world. It is behind many of the most popular games such as Crossy Road, Among Us, Angry Bird, Genshin Impact and a lot more. Moreover, it not only is well-suited for both 2D and 3D games but has also become a powerful tool for VR and AR development.

This series of Unity Game Development Program teach students some core techniques of developing both 2D and 3D games in Unity. It covers a wide range of topics from character control, coding (in C#), to asset management.

Program Outline

Game Genre (with classic examples)	Brief description of game (details subject to changes)	Concepts to be taught throughout the modules: - OOP application using C# - Game controls (keyboard/mouse/touch inputs) - UI design - Understanding game genre - Hierarchy in game design - Physics application - 2D/3D Vector for motion application - Orientation concept (rotation/degree/radian) - Animation - Collision handling - Visual effects - Sound effects - Automation - Crowd control - Character intelligence control - Multi-platform publishing (e.g. pc, android)
2D Platformer (e.g. Mega Man)	Controlling character to navigate on a 2D platform while performing actions such as attacking enemies and collecting rewards. Involves learning about character animation and control and asset applications.	
Skill Arcade (e.g. Stack-the-Box)	Placing of objects with precision and speed. Involves learning tricks to simulate object deformation in 3D environment.	

Program Outline		
Car Racing (e.g. Outrun)	Classic 3D car racing on randomly generated tracks. Involves learning about using codes to create changing environment to induce an element of surprise.	
Predator Arcade (e.g. Snake)	Character evolves by growing bigger as it eats food that is auto-populated on the terrain. Involves learning about dynamic character customisation and evolution as the game progresses.	
Shoot'em Up (e.g. Space Commander)	Controlling spacecraft and firing bullet streams at multiple enemies. Involves learning about generation and control of massive combat elements in a multi-enemy environment.	
First Person Shooter (e.g. Doom)	First-person view shooting game with multi-player. Involves multiple-angle controls and split views.	
Strategy (e.g. Tower defence)	Territorial defence through obstruction of enemies' advances. Involves building of allies and balancing of forces.	

THE LAB
LEARNING WITHOUT BOUNDARIES

Membership Fees

Exclusive Access

Elective Workshops at members' prices

Merchandise at members' prices

Weekly classes; 100 minutes per class

The Lab Python Programming

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

The Lab Unity Game Development

10 lessons	$850

<To complete 10 lessons in 12 weeks>

The Lab Competitive Programming

4 lessons/mth	$400

For General Enquiries:
Telephone: (+65) 8916-0017
Email: contact@thelab.sg

Central
The Centrepoint
176 Orchard Road
#03-18/19
Singapore 238843

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

Follow us