**OPAL EDUVENTURE**

**Tabs**

1. Home -

2. About Us

1. Mission
2. Vision
3. Goal -
4. Team
5. Pilot School
6. Partners
7. In News (Gallery)

3. What we do

1. Capability Development of Institute/School
2. Capacity Development of Teachers
3. Skill Development of Students(Personalization of Learning through Big Data)
4. Capacity Development of Parents
5. Computational Thinking

4. Code genius program

1. Pre School()
2. I
3. II
4. III
5. IV
6. V
7. VI
8. VII
9. VIII
10. IX
11. X

5. Events

6. School Mitra

1. SMS
2. Online Examination Portal

7. Contact us



**CODING IS THE NEW LITERACY**

We believe that every child should have the chance to code. There are many things that children will love about coding; creating their own game, app, making their own animated stories and the many other endless possibilities by harnessing today's rapidly evolving technologies.

**COURSE ROADMAP**

Course sequences are designed to offer clear and gradual progression

Age group 8 -12

1. Basics of code genius  
   Level 0  
   Introducing programming
2. Basics of code genius  
   Level 1  
   Starting with scratch
3. Basics of code genius  
   Level 2  
   Building with scratch
4. Basics of code genius  
   Level 3  
   Scratch for Robots
5. Basics of code   
   Level 4  
   Exploring Robotics programming in scratch
6. Basics of code genius   
   Level 5  
   Introduction to Python
7. Basics of code genius   
   Level 6  
   Introduction to Python 2
8. Basics of code genius  
   Level 7  
   Mobile MIT App inventor

Age Group 13 – 16

1. Genesis of code   
   Level 1   
   Python Fundamentals Through Minecraft
2. Genesis of code   
   Level 2  
   The Python session
3. Genesis of code  
   Level 3   
   Python Abstractions
4. Genesis of code  
   Level 4  
   Game Programming with PyGame

Continue with JAVA SCRIPTS or PYTHON

1. Genesis of code  
   Level 5  
   Algorithams
2. Genesis of code  
   Level 5J  
   Web programming with JavaScript
3. Genesis of code  
   Level 6  
   Machine Learning
4. Genesis of code  
   Level 6J  
   Networks and Servers
5. Genesis of code  
   Level 7J  
   Programming Mobile Apps

**Why is learning to code important?**

Programming (or coding) is the art of telling computers what to do. And as we go into the next wave of technological innovations, it is important, now more than ever, to equip our children with the skills necessary to navigate and create their future world.

In the past, programmers were trained in universities, honing their craft as they enter the workplace. However, the big changemakers who have made big strides in transforming computer technology often had their start early on in life or in their early teens. (Think Bill Gates, Mark Zuckerberg, and Elon Musk… and more)

Learning programming early develops the brain as the child gains a deeper understanding of logical thinking. By using computational thinking skills as they practice writing their programs, students enhance their ability in logical reasoning which will have an everlasting impact on their future in whichever career that they chose.

Over the next 10 years and beyond, programming is expected to be the fastest-growing occupation in the world. With 1 million jobs going unfilled in critical fields such as Big Data, Analytics, Artificial Intelligence and Robotics, we seek to provide our students with the foundation in logical thinking and problem solving that will be valuable for their success in their future world.

What if your child could learn skills that will allow them to write codes to program a website; or design a mobile app; or program a robot? What wonders would they create? At Computhink, we believe in making the learning process fun, interactive and purposeful. Students have a reason to learn what they deem as interesting and in that process, they are immersed in their personalized creative world and learning experience.

This is why the ---------- programme is so unique.

Our Impact

No. of game developed: 2000

Testimony from parents and students

**BASICS OF CODE: LEVEL 1**

PRIMARY 2-6 (AGES 8-12)

**STARTING WITH SCRATCH**

Basics of Code Level 1 is OPAL Eduventure’s foundational course. Students will be introduced to the fundamentals of coding through programming the world's favorite robot. Students will subsequently learn to program their own music, video games and animation using Scratch.



We will touch on the nature of computers - where they are found, why they are useful, and how they work. Code Campers will come to recognise code as a language used to instruct computers, as well as a means to tackle complex problems.

How do you use computer science to solve a problem? You start by learning how to formulate algorithms (chain tasks together to solve a problem), construct loops (computers are great at repetitive tasks so exploit that!) and rely on conditionals (enable your computer to make decisions). After this course, kids will be able to build basic programs and collaborate with others to devise computer-based solutions to problems.

**BASICS OF CODE: LEVEL 2**

PRIMARY 2-6 (AGES 8-12)

**BUILDING WITH SCRATCH**

Continuing from our foundational course, Basics of Code 2 is all about application and build experience. Our Code Campers will use their foundational computational tools learnt in Basics of Code 1 to produce more complex graphical animations, design multiple game components and compose their own mini-programs in Scratch.



The projects built in this course will require students to think mathematically (using kinematic and geometric concepts) and apply creative solutions logically (story boarding, how to nest loops and conditionals and write event-driven programs). After this course, Code Campers will have the beginnings of their coding portfolio and obtained greater literacy in Scratch to continue coding semi-independently.

**BASICS OF CODE: LEVEL 3**

**PRIMARY 2-6 (AGES 8-12)**



**BASICS OF CODE: LEVEL 4**

**PRIMARY 2-6 (AGES 8-12)**

**Scratch for Robots**

**BASICS OF CODE: LEVEL 5**

**PRIMARY 2-6 (AGES 8-12)**

**INTRODUCTION TO PYTHON**

No prior experience with Minecraft is needed for this course, though programming experience equivalent to having taken our ***Basics of Code: Level 4*** course is required. Please note that the focus of this course is to teach fundamental Python programming concepts through Minecraft as an example, and not to use Python to enhance gameplay in Minecraft.

The 5th course in the Basics sequence represents a major leap for our Code Campers as they migrate from the relative confines of [**Scratch**](https://en.wikipedia.org/wiki/Scratch_(programming_language)) and **[AppInventor](http://appinventor.mit.edu/explore/" \t "_blank)** into real-world programming with one of the world’s most popular programming languages – [**Python**](http://www.nature.com/news/programming-pick-up-python-1.16833).



Moving from the drag-and-drop environment of the first three Basics of Coding courses into a text-based environment represents a challenge but this is well worth the effort. With a fully featured language like Python, what Code Campers can do is limited only by their imagination as mobile, web development, artificial intelligence, cloud computing, robotics, financial engineering and bioinformatics are well within reach of the savvy Python programmer.

To help familiarise Code Campers with the basics of programming in the Pythonic context, we have chosen to elucidate these concepts through **[Minecraft](https://minecraft.net/" \t "_blank)** – one of the most iconic, beloved and dare we say, educational (so declares the [**New York Times**](http://www.nytimes.com/2016/04/17/magazine/the-minecraft-generation.html)) video games of our time.

**BASICS OF CODE: LEVEL 6**

**PRIMARY 2-6 (AGES 8-12)**

**INTRODUCTION TO PYTHON 2**

No prior experience with Minecraft is needed for this course, though programming experience equivalent to having taken our ***Basics of Code: Level 5*** course is required. Please note that the focus of this course is to teach fundamental Python programming concepts through Minecraft as an example, and not to use Python to enhance gameplay in Minecraft.

This final course in the Basics sequence continues our exploration of computer programming with one of the world’s most popular programming languages – [**Python**](http://www.nature.com/news/programming-pick-up-python-1.16833).

Students learn how a programming language grants them access to basic computer capabilities - data storage (variables), machine repetition (loops) and logical reasoning (conditionals). Through in-class coding exercises in a lab environment, they then learn first-hand how these capabilities can be harnessed to chip away at almost any generic real-world problem to yield a solution.

Students graduating from this course may choose to proceed on to our more advanced offerings by next taking [***Principles of Code Level 2***](http://www.sgcodecampus.com/courses/poc2/).



**Age Group 13 – 16**

**GENESIS OF CODE: LEVEL 1**

**SECONDARY 1 - 4 (AGES 13-16)**

**PYTHON FUNDAMENTALS THROUGH MINECRAFT**

No prior experience with programming or Minecraft is needed for this course. Please note that the focus of this course is to teach fundamental Python programming concepts through Minecraft as an example, and not to use Python to enhance gameplay in Minecraft.

This first course of our **Principles of Code** sequence introduces our Code Campers to the vocabulary and grammar of coding in the [**Python**](http://www.nature.com/news/programming-pick-up-python-1.16833) language - the fundamental concepts that they must master before moving on to build computer programs that solve practical problems or create real-world apps. With a full-feature language like Python, what Code Campers can do is limited only by their imagination as mobile, web development, artificial intelligence, cloud computing, robotics, financial engineering and bioinformatics are well within reach of the savvy Python programmer.



To help familiarise Code Campers with the basics of programming in the Pythonic context, we have chosen to elucidate these concepts through **[Minecraft](https://minecraft.net/" \t "_blank)** – one of the most iconic, beloved and dare we say, educational (so declares the [**New York Times**](http://www.nytimes.com/2016/04/17/magazine/the-minecraft-generation.html)) video games of our time.

**GENESIS OF CODE: LEVEL 2**

**SECONDARY 1 - 4 (AGES 13-16)**

**THE PYTHON SESSIONS**

This course is designed to serve as a bridging course for graduates of our [**Basics of Code**](http://www.sgcodecampus.com/#boc-rm) sequence (for ages 8-12) transiting into more advanced programming as well as an optional course for [**Principles of Code Level 1**](http://www.sgcodecampus.com/courses/poc1/) students who wish to spend more time learning the intricacies of elementary Python programming before hitting the more abstract ideas covered in [**Level 3**](http://www.sgcodecampus.com/courses/poc3/).

Classes are run as a guided problem solving sessions where students apply their newly acquired Python skills to solving puzzles in text processing, graphics programming and simple game development.



**GENESIS OF CODE: LEVEL 3**

**SECONDARY 1 - 4 (AGES 13-16)**

**PYTHON ABSTRACTIONS**

In the physical world, many people solve problems through analogy - rehashing and reusing solutions to old problems we have seen to new problems that look and feel familiar.



In computer science, we study how to do this through what computer scientists call **abstraction**. Code Campers will learn to bend the Python language to fit the context of the problems they are solving. This approach is what enables financial coders to create programmatic representations of stocks and bonds to model market prices, and bioinformatics researchers to code up the human genome to study its properties.

**GENESIS OF CODE: LEVEL 4**

**SECONDARY 1 - 4 (AGES 13-16)**

**GAME PROGRAMMING WITH PYGAME**

While the earlier courses were primarily focused on providing our students with the foundation and the core ideas defining the craft of computer programming, this 4th course can be seen as the programming equivalent of a creative writing class.



[**PyGame**](https://en.wikipedia.org/wiki/Pygame) is a lightweight albeit powerful game engine that allows students to build and examine the internal workings of some of their favourite video games. Programming concepts hitherto learned like loops, functions, arrays, hashes and object oriented programming come to life as students design and build their way through a generic **[platformer](https://en.wikipedia.org/wiki/Platform_game" \t "_blank)** and a [**2D RPG**](https://en.wikipedia.org/wiki/Role-playing_video_game) game.

**GENESIS OF CODE: LEVEL 5**

**SECONDARY 1 - 4 (AGES 13-16)**

**ALGORITHMS**

An **algorithm** is defined as a sequence of steps required to solve a given problem. In the next step of the Code Campus learning journey, we concern ourselves with the **analysis of algorithms** - a careful, rigorous, hands-on study of how to improve our solutions. What makes one solution better than another? How is that improvement quantified?



As solving a problem almost always involves an imposition of structure upon chaos, our foray into this topic begins with a hard look at **sorting algorithms** - ways to put data in order. When asked for “the most efficient way” to order a million numbers while on a visit at Google, Barack Obama famously remarked “I think the **Bubble sort**would be the wrong way to go.” In this, the American president was right - astonishingly, moving from a [**Bubble sort**](https://en.wikipedia.org/wiki/Bubble_sort) to a **[Quicksort](https://en.wikipedia.org/wiki/Quicksort" \t "_blank)**reduces the time taken to sort by a factor of tens of thousands.

We will also study how shrewdly structuring data is often a big chunk of problem solving - how does thinking about the internet or the retail economy in terms of networks and graphs allow us to better model the flow of information or trade?

In the previous courses, we learned how to code - in this course, we will learn, quite simply, **how to code better**.

**GENESIS OF CODE: LEVEL 5J**

**SECONDARY 1 - 4 (AGES 13-16)**

**WEB PROGRAMMING WITH JAVASCRIPT**

At Code Campus, we think that learning to program the web is a challenging but ultimately rewarding endeavor because there are few skills truly as enabling as being able to create a web application. Our approach to web programming is decidedly opinionated in that it assumes that students have already acquired the basic skills needed to read and write computer programs before learning how those skills can be applied to the web domain. [***Principles of Code Level 3***](http://www.sgcodecampus.com/courses/poc3/) is thus a hard requirement for this offering.



Programming the web can broadly be decomposed into three connected tasks: structuring content, designing the appearance of said content, and programming applications based on that content. Learning to build on the web can thus be reduced to learning about the associated technologies which address these concerns - HTML, CSS and JavaScript.  
The course will first start with a primer of the JavaScript programming language and compare it to other popular languages students are familiar with like Python, before delving into the sister technologies of HTML and CSS. At the end of the course, Code Campers will be able to build their own landing pages and simple frontend web applications through the use of popular web programming libraries like [jQuery](https://jquery.com/" \t "_blank) and [Bootstrap](http://getbootstrap.com/)

**PRINCIPLES OF CODE: LEVEL 6**

**SECONDARY 1 - 4 (AGES 13-16)**

**MACHINE LEARNING**

Computers are advancing so rapidly that researchers are already predicting that we will encounter machines as intelligent as human beings within [**the next decade**](http://www.dailymail.co.uk/sciencetech/article-3408366/Here-come-robots-Davos-bosses-brace-big-technology-shocks.html). 2016 was a milestone year for Artificial Intelligence that saw a [**Google AI beat a world champion**](http://www.nytimes.com/2016/03/16/world/asia/korea-alphago-vs-lee-sedol-go.html) in the game Go, a game that most acknowledge is several orders more difficult than chess.



How do we navigate a world where robots and AIs are intertwined with our lives and the workplace? Code Campus believes that developing an appreciation and basic understanding of this technology is a must for preparing our students for jobs that do not yet exist in a world, where a human response is increasingly becoming indistinguishable from a machine’s.

We will use Python’s excellent **[Scikit-Learn](http://scikit-learn.org/stable/" \t "_blank)** library to study how basic machine learning algorithms like the k-means, that only depend on the basic knowledge of an arithmetic average, can be used to endow computer programs with simple intelligence. We will also study how matrix math underlies the storied PageRank algorithm - the secret sauce for Google’s early success in web search, as well as the principles behind of how sites like Amazon serve up recommendations to their online customers.

Due to the relatively more advanced mathematical requirements of this course, prospective students will be required to be at least at the Secondary 3 level and will have to go through a verbal aptitude test.

**GENESIS OF CODE: LEVEL 6J**

**SECONDARY 1 - 4 (AGES 13-16)**

**NETWORKS AND SERVERS**

Trying to make sense of how computers communicate, or even the recent phenomenon known as the [**Internet of Things (IoT)**](https://www.theguardian.com/technology/2015/may/06/what-is-the-internet-of-things-google) begins with the an understanding of the humble **server** - how these are built with code and responsibly secured for users.



In this course, Code Campers will first lear n to use Node.js to program a variety of physical sensors connected to a server hosted on a [**Raspberry Pi**](https://www.raspberrypi.org/help/faqs/#introWhatIs) - a powerful credit card sized computer that retails for less than SGD50.00. Next, they will learn the rudiments of network programming and computer security in JavaScript and how to set up their own cloud computing resources and servers using services like **[Heroku](https://www.heroku.com/" \t "_blank)**. The class will then conclude with Code Campers building their very own web-accessible home surveillance system.

**GENESIS OF CODE: LEVEL 7J**

**SECONDARY 1 - 4 (AGES 13-16)**

**PROGRAMMING MOBILE APPS**

As the number of mobile phone users grows exponentially year after year, we now awaken to a world that is [**increasingly accessed, run and driven by mobile**](https://www.bcgperspectives.com/content/articles/telecommunications_connected_world_growth_global_mobile_internet_economy/?chapter=2). In this course, Code Campers will learn how to use the JavaScipt framework [**React Native**](https://facebook.github.io/react-native/showcase.html) developed by Facebook to create performant mobile apps that can be delivered to both iOS and Android phones, giving them a global platform upon which to showcase their digital creations.



Code Campers will learn how to utilise JavaScript tools that access information on the web, create beautiful user interfaces, build databases that store the data consumed by their apps and deploy their creations into the mobile app marketplace.