Computer Programming 10-12

At WSS we encourage students to try computer programing (regardless of their age or grade level). This creates a diverse Computer Programming class that must accommodate a wide variety of skill and experience level. **But don't worry…we have you covered!**

To help start you off on the right path, please consider the following course levels:

Level -1

Introduction to Computer Programming:

- Python
- Robot-C (Robotics)
- Intro to web programming
- If it's your first experience computer programming, then start here (Level 1)

Level 2

Advanced Python/Web Programming/Robotics

- Advanced lessons, exercises and projects in Python.
- Web Programming including JavaScript
- Intermediate Robotics problem solving.

Level 3

Career/Post Secondary Prep

- Super Advanced Python
- Work on curriculum appropriate to your future studies/work in computer programming
  Intro to some essential Computer Scinece Principles
- Grade 11 or 12 Credit
- Curriculum is continuously expanding!
- Robotics/Web Programming option



Level 1 Computer Programming

Welcome to your new world of Computer Programming. This is an exciting opportunity for you learn the basics of how **software** is made and introduce you to coding.

This course aims to give you the ideas and experience necessary to solve problems, create applications, and build games using a computer. Level 1 we will explore these areas by learning one of the most commonly used and powerful computer programming languages available - **Python!** Python is one of the easiest languages to get started with and has all the tools to help you learn essentials skills in programming.

You will also be introduced to Robotics programing with Lego Mindstorms and the RobotC language.

Students will also get a brief introduction to Web programming.

Topics will include:

- What is Computer Programming?
- Programming methodology
- Understanding using common programming tools to solve problems
  (Input/output, Decision making, Logic, Loping, Repeating).
- Algorithm development
- Robotics programming. Syntax and common rules for the program language "C"
- Basics syntax and rule for Python®
- Basics of Web programming

Marking Scheme:

Assignments and Projects:	60%
Tests and Quizzes:	20%
Exam:	10%
Core Competencies/Self-Assessment:	10%

Ask about the University of Waterloo Computer **Programming Contest**.

Fun and easy! No experience necessary!

Level 2

Advanced Programming Topics Using Python

Students in their 2rd year of Computer Programming will be asked to gain more experience using python to solve **challenging problems** and **creating user applications**.

- **Problem solving and Algorithm development**
- **Nested For loops**
- **2-D List (Array) manipulation**
- **Sorting**
- **Creating more complex and cool applications**

Web Fundamentals

Hey let's face it, tons of software involves the web. In this section you will learn the building blocks of web development with HTML and CSS, and JavaScript to create specifically designed web pages and interactive applications. Assignments, resources, lessons, will be taken from carefully selected online platforms.

Intermediate Robotics Challenges

You'll be asked to test your programming skills to solve intermediate robotics problems while using Lego Mindstorms and the programming language RobotC.

Students who have taken a prior computer programming course will be **strongly** encouraged to participate in the University of Waterloo Computer Programming contest **February.** See course Website: www.walzl1.com for more details of Waterloo Contests.

Marking Scheme:

Assignments and Projects:	60%
Tests and Quizzes:	10%
Exam:	10%
Core Competencies/Self-Assessment:	20%

Ask about the University of Waterloo Computer **Programming Contest**.

Fun and easy! No experience necessary!

Level 3 - Super Advanced Python

All level 3 students will be **strongly** encouraged to prepare for and participate in the University of Waterloo Computer Programming contest in February. See course Website: www.walzl1.com for more details of Waterloo Contests.

Successful completion of level 3 will include a good understanding of the following topics:

- **Split and slicing lists**
- **List comprehension**
- **Incorporating functions effectively within larger programming**
- **Tuples**
- **Sets**
- **Dictionaries**
- **Recursion**
- **Fundamentals of Object- Oriented programming:**
  Classes, method, attributes.
  Inheritance
  Practice using OOP to solve problems
- **2-D arrays**
- **Sorting Algorithms**

Marking Scheme:

Assignments and Projects:	60%
Tests and Quizzes:	10%
Exam:	10%
Core Competencies/Self-Assessment:	20%

Post Secondary Preparation

If you are have finish Level 3 good for you! You have probably been programming for 3 years and/or have mastered many challenging topics.

Your job will be to find and complete course or curriculum that you think will prepare you for YOUR future exploration in computer programming.

Working with their instructor student will be asked to create a learning plan based on a course or workplace/school environment they plan to enter after graduation.

In previous classes, students have networked with BCIT, University of Toronto, University of Calgary, University of Waterloo, UBC, and Information technology companies to discover and learn material that will help prepare them for their future.

Getting Help

If you need some extra help, please don't hesitate to ask. I will make myself available outside class time if you need any further explanation. Tutorials will be held **Monday's and Wednesdays** afterschool.

Absences

Most of the learning and evaluation in this course will be based upon in-class assignments and projects. Attendance is mandatory. Students must complete missed work and demonstrate an understanding of course material to receive a passing grade. If you are away, you have two options for catching up: Using the lab at lunch or afterschool. Signing out equipment to take home. **It is the students responsibility to catch up on any material they miss doing an absence.**

Classroom Expectations:

To ensure a safe and effective learning environment it is expected that students abide by the following rules:

1. **Respect others**. **Help others. Be kind. Be caring**. Make sure you are on time for class; late arrivals can disrupt lessons, activities, or discussions. **If you need help or have a question,** raise your hand **or approach Mr. Walzl's desk politely.** You are encouraged to ask questions and seek help; However, computer programming will require you to problems solve and overcome challenges on your own and work with your classmates.

2. **Respect the Lab**. While you are in the lab, it is important to remember that equipment in the room is valuable and can be easily misplace or broken. **Make sure you leave time at the end of class to clean up your equipment and put it away in an organized fashion.**

3. **Focus**. Remember you are in school to learn and grow. While in class**, work hard to understand and enjoy the material we are covering**. Computer Programming can be a lot of fun.

Keys to Great Learning and Good Grades:

1. Use class time appropriately, you are here to learn so always keep that in mind.☺

2. Use the internet as a resource. Wow I can't emphasise this enough. There are thousands Solutions, videos, sample code ready for you to find!...Have a problem…google it! ☺

3. Get involved in class. Ask lots of questions (always ask if you don't understand). **Be patient**…it can sometimes take a long time to work out a "bug" in a piece of code….all part of the process.

4. Ask a friend. Use neighbors and friends to help you; However, don't just take their code or let them code for you…it's hard to learn anything this way.

How Am I Doing?

(Core Competencies)

In addition to the technical **skills** and **knowledge** you learn in Computer Programming you are also expected use your skill and knowledge in the following ways:

a) Work effectively with others
b) Create stuff
c) Displaying your work
d) Solve problems

If you can check off the items above, you are on the right track! But wait, it's not that easy. If you are really pushing yourself to **grow, develop, and learn** there are a few more ingredients that make for positive development in this course:

Time and Effort

Learning anything new takes time and energy. How hard are you working? What challenges have you overcome? How much time are you spending on your project?

Organization

Do you know what your goal is? Do you have a plan? Do you know when you want certain things to be done?

Connections

Working with others can be an effective way to learn or assess your work. Have you shown your work to others? Shared your ideas and plans? Have you helped a classmate with their work? Contacted an expert? Have you asked for help? Have you listened to others?

Your Satisfaction.

Ultimately you are the boss. Learning isn't always easy, but you should, ultimately, be happy with what you are doing and where you are going. If you are regularly feeling good about your work and progress, you are bound to be on the right track.

	Needs improvement	Satisfactory	Great
Working with well with others?	I am not open to working with others. I rarely share my work or ask other for help.	I work with others in class. I regularly share my work, ask for help, and help others.	I share my work. I help others and they help me. I am a positive influence when I work with another individual or group. I can think of instances where I have **worked together** with others solve to a problem.
Creating?	I am learning stuff, but not usually creating anything new. I like to ask for and follow detailed instructions regularly.	I am definitely learning. I am sometimes able to put ideas together create something new or different.	I discover and create new things in my work regularly. My work usually **exceeds the expectations** of a given assignment or project.
Displaying Work?	People are too busy to see my work. There have been a few projects where people did not see my work	I show my work to my classmates and Mr. Walzl only when instructed to do so.	I'm proud of my work and display it in a way that people understand. I always make sure to **show my projects to classmates and Mr. Walzl**. I sometimes show my work to people outside of class
Solving Problems	When I get stuck. My progress slows down.	When I get stuck I actively look for new options and solutions. I make a plan to move forward	When I get stuck, I examine all options, make a plan and follow through until the problem is solved. I am able to work with others effectively and positively while solving problems
Time and Effort	I don't always spend time in class effectively. I am off task frequently and distract others	Time spent in class is ***always*** affective.	Time spent in class is **always** affective for myself and others. I spend time **outside class working on projects and problems.**
Organization	I usually know what needs to be done in class and I have a rough idea of how to do it.	I know exactly what needs to be done and I have a plan of how to do it.	I always have a direction and plan for my work. Dates and events have been recorded. Work is documented so progress and quality can be examined.
Satisfaction	I work in a way that isn't always rewarding.	I work in a way that allows me to enjoy my work in class	I make an effort to enjoy each class and make it valuable for myself and others.