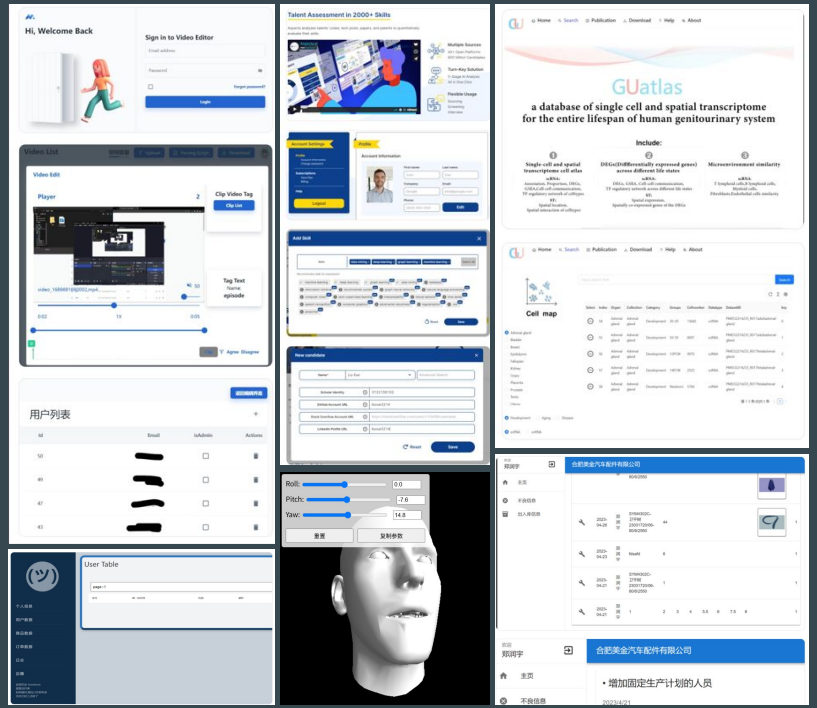
AP Computer Science A

Trial Lesson 1

Yunu Jung 정윤우

1. Computer Science at Sichuan University
2. Full-stack developer
3. Server Maintainer
4. Coding for hobby



Introduction

What Is AP Computer Science A?

- College-level programming course using Java
- Focuses on writing, analyzing, and debugging code
- Helps build strong problem-solving and logical thinking skills
- Prepares students for further studies in computer science

What are we going to learn in this course?

- Java syntax and basic programming structures
- Control flow logic
- Object-oriented programming (OOP)
- Working with arrays and ArrayLists
- Computational thinking and problem-solving
    - Breaking problems into smaller parts
    - Designing step-by-step algorithms
    - Thinking like a computer to find efficient solutions

Where is it used in real life?

- University Projects – Data Analysis and Research
  - Psychology, Economics, Biology
- Problem Solving – Thinking Like a Programmer
  - Structured, Logical thinking that helps solve problems
- Understanding and Collaborating with AI
- Becoming a Software Engineer or Tech Professional

AP CSA Exam Overview

Exam

Section	Question Type	Number of Questions	Exam Weighting	Timing
I	Multiple-choice questions	40	50%	90 minutes
II	Free-response questions	4		90 minutes
	Question 1: Methods and Control Structures (9 points)		12.5%	
	Question 2: Class (9 points)		12.5%	
	Question 3: Array/`ArrayList` (9 points)		12.5%	
	Question 4: 2D Array (9 points)		12.5%	

The exam assesses content from the three big ideas for the course:

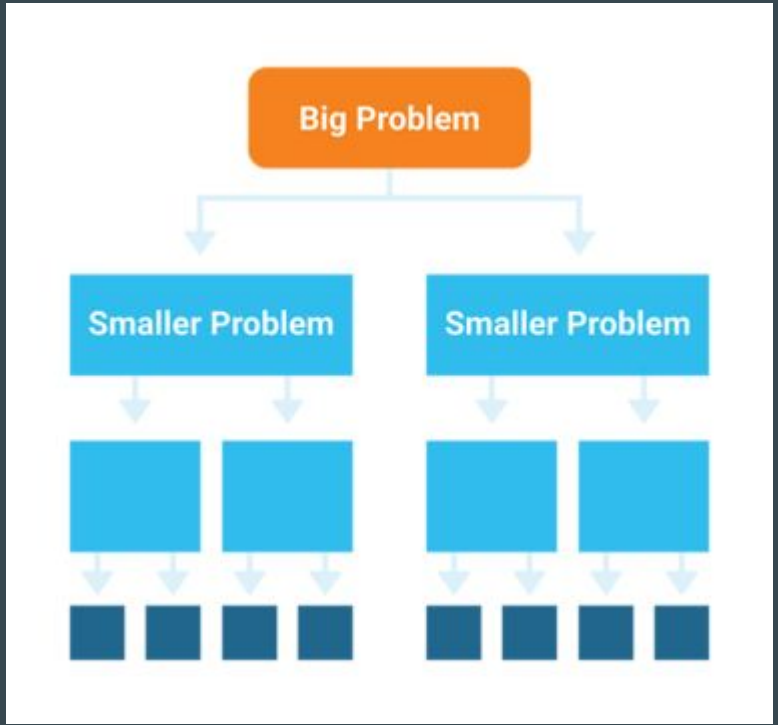
Big Idea 1: Modularity

Big Idea 2: Variables

Big Idea 3: Control

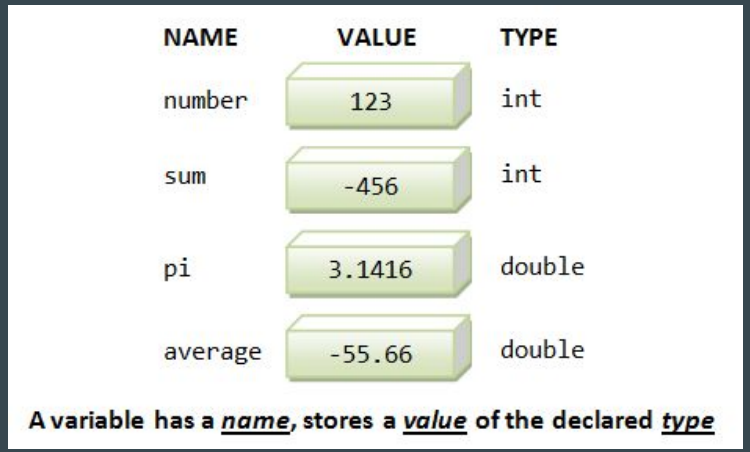
BIG IDEA 1: MODULARITY

Incorporating elements of abstraction, by breaking problems down into interacting pieces, each with their own purpose, makes writing complex programs easier. Abstracting simplifies concepts and processes by looking at the big picture rather than being overwhelmed by the details. Modularity in object-oriented programming allows us to use abstraction to break complex programs down into individual classes and methods.



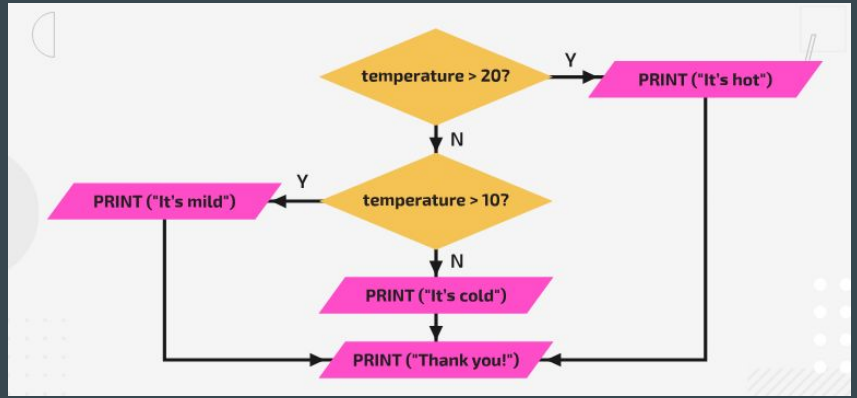
BIG IDEA 2: VARIABLES

Information used as a basis for reasoning, discussion, or calculation is referred to as data. Programs rely on variables to store data, on data structures to organize multiple values when program complexity increases, and on algorithms to sort, access, and manipulate this data. Variables create data abstractions, as they can represent a set of possible values or a group of related values.



BIG IDEA 3: CONTROL

Doing things in order, making decisions, and doing the same process multiple times are represented in code by using control structures and specifying the order in which instructions are executed. Programmers need to think algorithmically in order to define and interpret processes that are used in a program.



Course

Units	Exam Weighting
Unit 1: Primitive Types	**2.5–5%**
Unit 2: Using Objects	**5–7.5%**
Unit 3: Boolean Expressions and `if` Statements	**15–17.5%**
Unit 4: Iteration	**17.5–22.5%**
Unit 5: Writing Classes	**5–7.5%**
Unit 6: Array	**10–15%**
Unit 7: `ArrayList`	**2.5–7.5%**
Unit 8: 2D Array	**7.5–10%**
Unit 9: Inheritance	**5–10%**
Unit 10: Recursion	**5–7.5%**

MCQ 1

1. Evaluate the following expression: `4 + 6 % 12 / 4`

(A) `1`
(B) `2`
(C) `4`
(D) `4.5`
(E) `5`

MCQ 2

2. Which of the following expressions does NOT evaluate to `0.2`?

(A) `(1.0 * 2) / (1.0 * 10)`
(B) `2.0 / 10`
(C) `(double) 2 / 10`
(D) `(double)(2 / 10)`
(E) `Math.sqrt(4) / Math.sqrt(100)`

MCQ 3

3. Choose the code used to print the following:

```
"Friends"
```

(A) `System.out.print(""Friends"");`

(B) `System.out.print("//"Friends//"");`

(C) `System.out.print("/"Friends/"");`

(D) `System.out.print("\"Friends\"");`

(E) `System.out.print("\\"Friends \\"");`

FRQ

1. This question simulates birds or possibly a bear eating at a bird feeder. The following `Feeder` class contains information about how much food is in the bird feeder and simulates how much food is eaten. You will write two methods of the `Feeder` class.

```
public class Feeder
{
   /**
    *  The amount of food, in grams, currently in the bird feeder; initialized in the constructor and
    *  always greater than or equal to zero
    */
   private int currentFood;

   /**
    *  Simulates one day with numBirds birds or possibly a bear at the bird feeder,
    *  as described in part (a)
    *  Precondition: numBirds > 0
    */
   public void simulateOneDay(int numBirds)
   {  /* to be implemented in part (a) */  }

   /**
    *  Returns the number of days birds or a bear found food to eat at the feeder in this simulation,
    *  as described in part (b)
    *  Preconditions: numBirds > 0, numDays > 0
    */
   public int simulateManyDays(int numBirds, int numDays)
   {  /* to be implemented in part (b) */  }

   // There may be instance variables, constructors, or methods that are not shown.
}
```

FRQ Answer

A Feeder class for simulation(1 var, 2 function)
public class Feeder
	private int currentFood
	public void simulateOneDay(int numBirds)
	public int simulateManyDays(int numBirds, int numDays)

Requirements
write function public void simulateOneDay(int numBirds)
no return only calculate and change private int currentFood
95% birds eats, 5% bear eats
each bird eats (10, 50] determine by each day
bear eat all

Points
example doesn't show the bear part and possibility, but need to implement
Math.random() < 0.05 test if bear come
(int) (Math.random() * 41 + 10) get each consum
[0, 1) * 41 = [0, 41)
[0, 41) + 10 = [10, 51)
(int)[10, 51) = only integer, range from [10, 50]

Logic
given birds
if 95%
	consum = random(10, 50]
	totalConsum = consum * birds
	currentFood = currentFood – totalConsum
	if currentFood < 0 then currentFood = 0
else
	currentFood = 0

(a)

```
public void simulateOneDay(int numBirds)
{
   double condition = Math.random();
   if (condition < 0.05)
   {
      currentFood = 0;
   }
   else
   {
      int eachBirdEats = (int) (Math.random() * 41) + 10;
      int totalEaten = numBirds * eachBirdEats;
      if (totalEaten > currentFood)
      {
         currentFood = 0;
      }
      else
      {
         currentFood -= totalEaten;
      }
   }
}
```

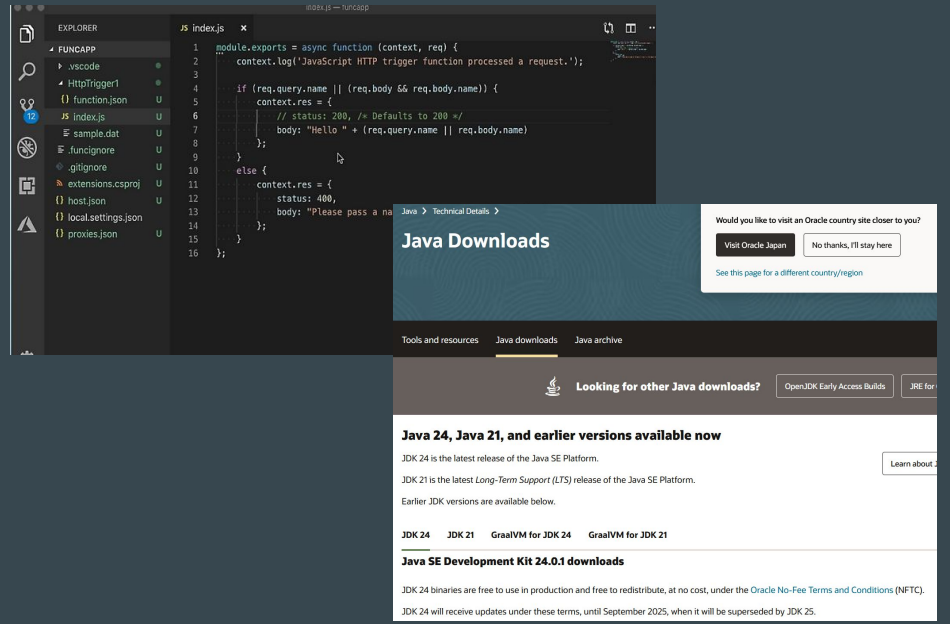
Practical Setup + Hands-on Coding

Why?

- Many students fail the FRQ because they're too afraid to try coding on their own.
- Practice makes the difference — we'll write code together, not just watch.

You need

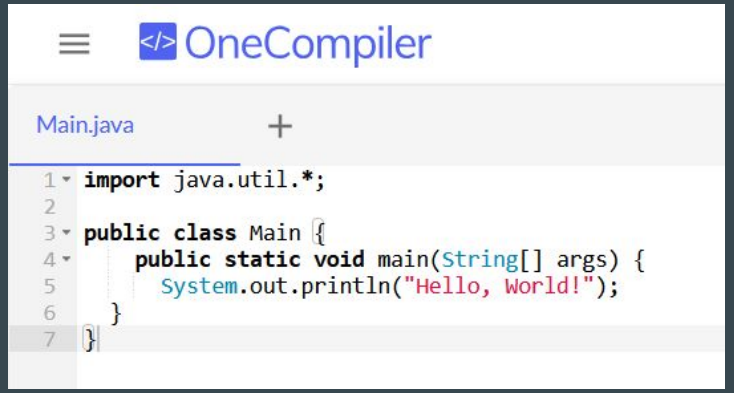
- Windows (or MacOS)
- IDE (VSCode)
- Java SDK

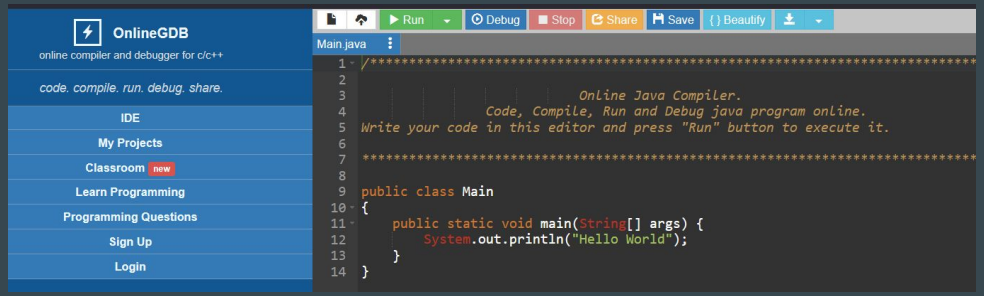


or.....

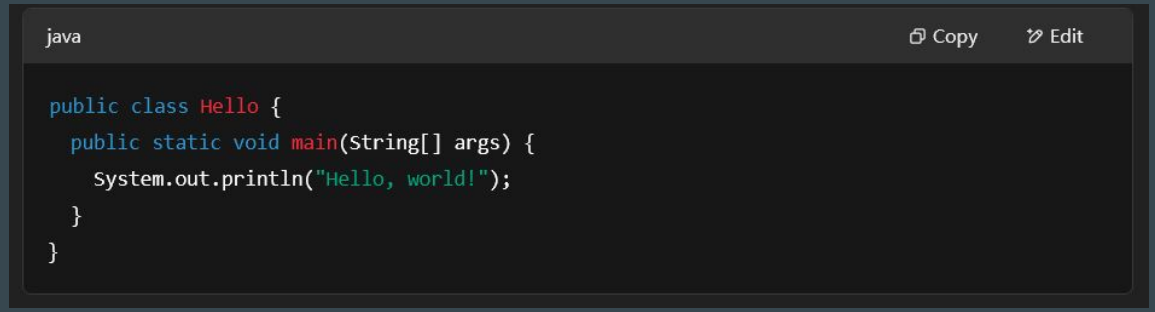
Or.....


Programiz
Online Java Compiler
Main.java
1 // Online Java Compiler
2 // Use this editor to write, compile and run your Java code online
3
4 class Main {
5 public static void main(String[] args) {
6 System.out.println("Try programiz.pro");
7 }
8 }


OneCompiler
Main.java
1 import java.util.*;
2
3 public class Main {
4 public static void main(String[] args) {
5 System.out.println("Hello, World!");
6 }
7 }


OnlineGDB
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
Run
Debug
Stop
Share
Save
{ } Beautify
Main.java
/*
Online Java Compiler.
Code, Compile, Run and Debug java program online.
Write your code in this editor and press "Run" button to execute it.
*/
public class Main
{
public static void main(String[] args) {
System.out.println("Hello World");
}
}

Hello World!


java
Copy
Edit
public class Hello {
public static void main(String[] args) {
System.out.println("Hello, world!");
}
}

What Did We Learn Today?

- Explored what AP Computer Science A is
- Wrote first real Java program
- 3 Big Ideas of the course
  - Modularity – Break problems into methods
  - Variables – Store and manage data
  - Control – Use logic to make decisions and loops