香港電腦教育學會
The Hong Kong Association for Computer Education

Experience sharing on teaching algorithm and programming

Jacky Hui Chun Kit
ICT Panel Chairperson
Carmel Secondary School

Major Change from Elective D to Elective C

Elective Part (Content) Option C (Algorithm and Programming)

a. Programming

b. ~~Programming Languages~~

c. ~~Systems Development~~

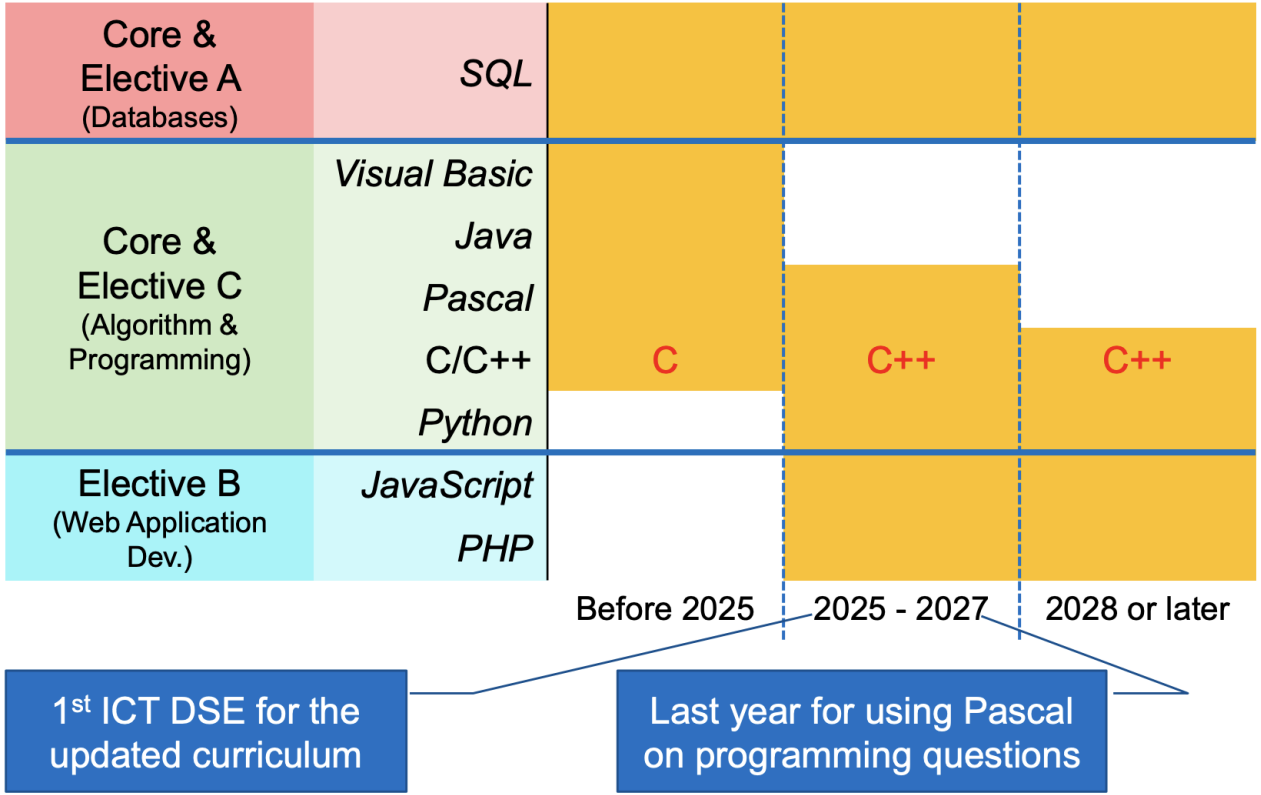
d. Applications of Programming in Real Life

https://www.edb.gov.hk/attachment/en/curriculum-development/kla/technology-edu/resources/computer-edu/Curriculum_Renewal_on_SS_ICT_20191205.pdf

Applications of Programming in Real Life (6 hours)

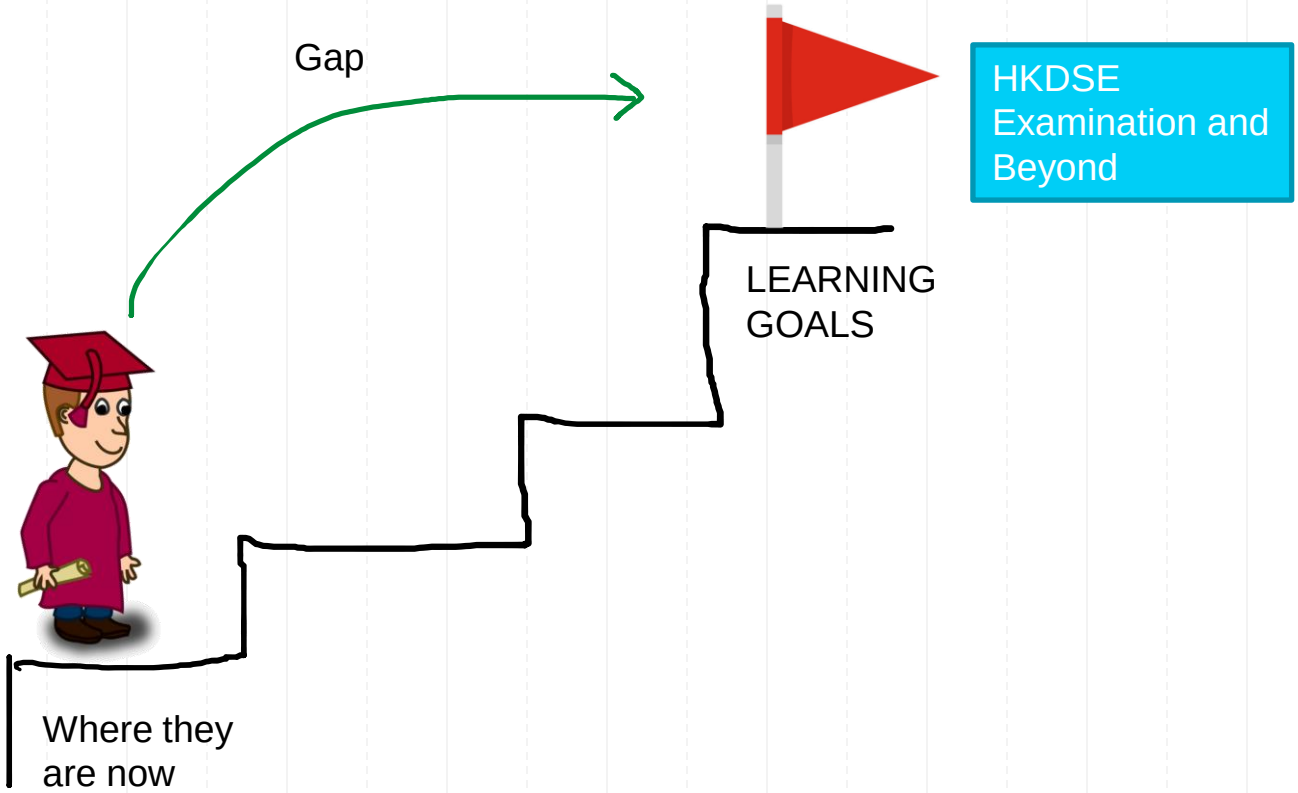
Learning Outcomes	Remarks
Use extended programming modules or libraries in writing programs to interact with physical devices.	Students should be able to use extended modules or libraries for capturing data from sensors (e.g. light sensor and accelerometer) and controlling specific devices (e.g. motor). Details of extended modules or libraries are not required.
Use event handlers in writing event-driven programs.	Specific events include user actions (e.g. pressing a button) and sensor values (e.g. the reading from the light sensor is over a defined value). Details of event handlers are not required.
Construct simple programs on physical devices by using features/components of physical devices like speech recognition and accelerometer.	Examples include generating a text display by speech recognition, controlling the movement of motors and detecting motion by accelerometer.

Programming Languages

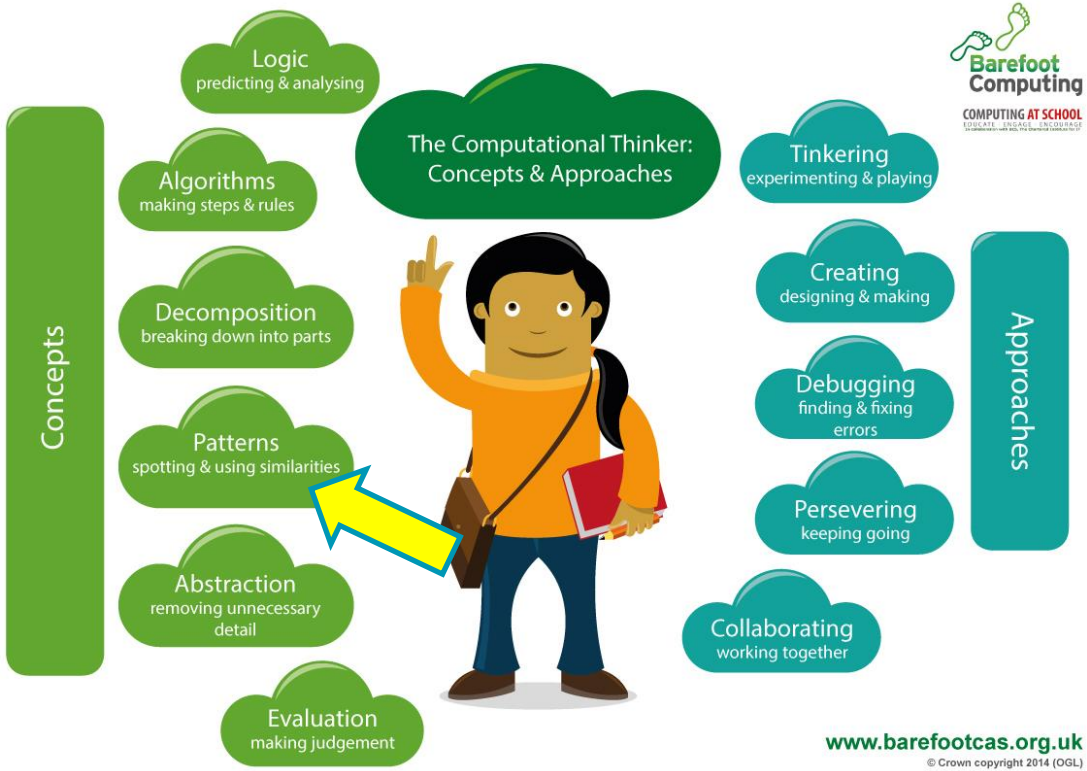


Useful Tools

https://sites.google.com/carmelss.edu.hk/edbhkaceshari
ng/home


Gap
HKDSE Examination and Beyond
LEARNING GOALS
Where they are now

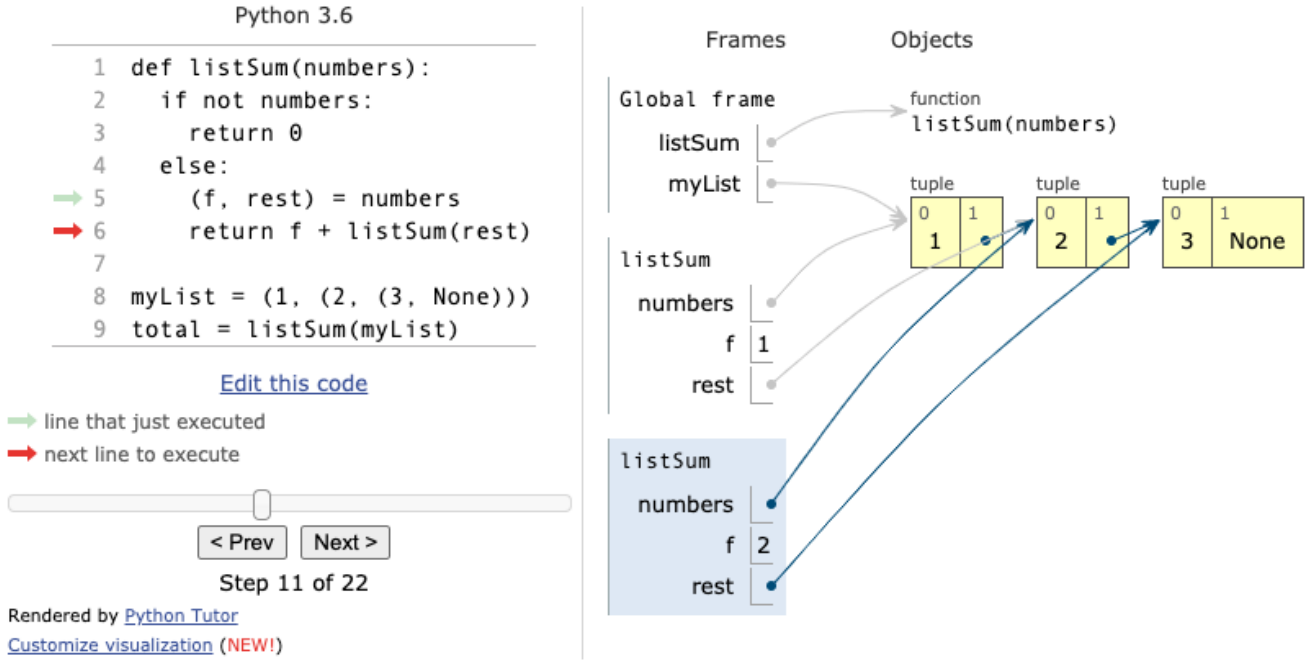
Teaching Content



Barefoot would like to acknowledge the work of Julia Briggs and the eLIM team at Somerset County Council for their contribution to this poster.

Visualizing Code Execution

You can also embed visualizations into any webpage. Here is a Python example:



Handling Errors

main.py

```
1  status = int(input())
2  if status == 0
3      print("Not OK")
4  else:
5      print("OK")
6
```

Console Shell Markdown

```
  File "main.py", line 2
    if status == 0
                 ^
SyntaxError: invalid syntax
```

3. Ricky is writing a program in Java. He saves the program source file using the filename program1.java. The following shows the contents of the file he is working on.

Line	Statement
01	`class myprogram {`
02	`    public static void main(String args[]) {`
03	`        System.out.println("Hello World!");`
04	`    }`
05	`}`

(a) (i) What is the command that he should type to compile the program?

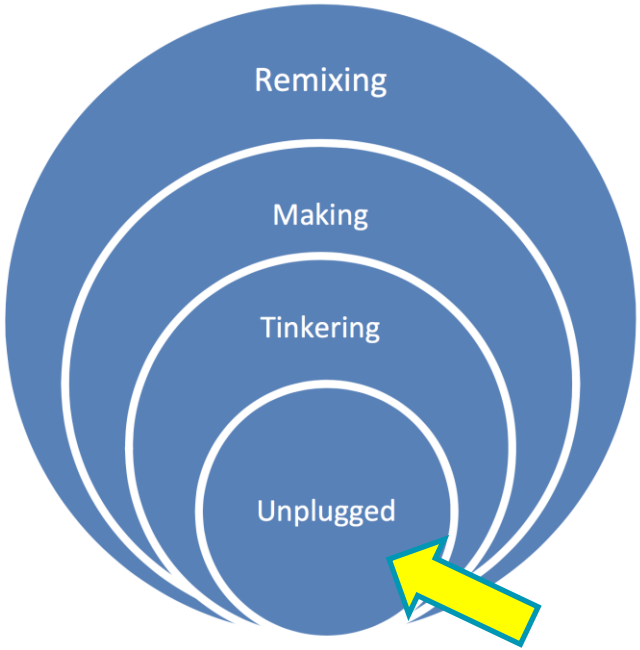
(ii) After compiling the program, what is the command that he should use to execute the program?

(2 marks)

(b) Ricky further modifies the program shown above. He encounters the following errors when he tries to compile the program.

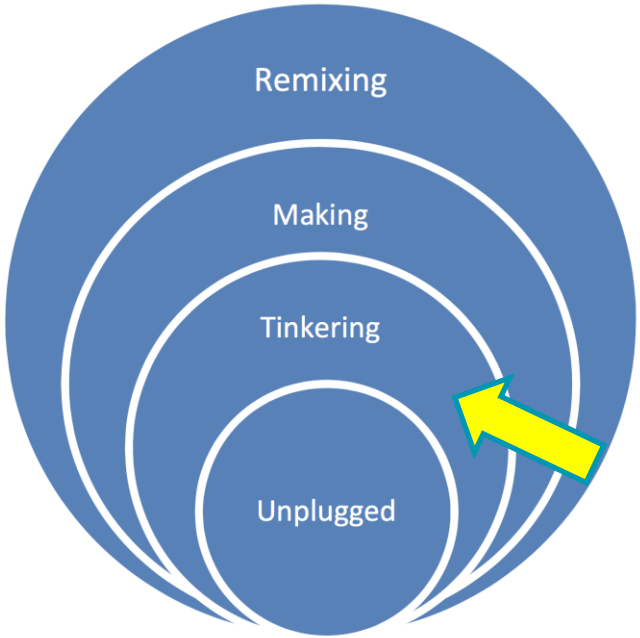
```
Main.java:5: error: cannot find symbol
     Scanner sc = new Scanner(System.in);
     ^
  symbol:   class Scanner
  location: class myprogram
Main.java:5: error: cannot find symbol
     Scanner sc = new Scanner(System.in);
                      ^
  symbol:   class Scanner
  location: class myprogram
Main.java:6: error: cannot find symbol
     b = 1;
     ^
  symbol:   variable b
  location: class myprogram
Main.java:10: error: incompatible types: possible lossy conversion
from double to int
     a = c;
         ^
Main.java:12: error: cannot find symbol
     System.out.printline();
                ^
  symbol:   method printline()
  location: variable out of type PrintStream
5 errors
```

CT Pedagogical Framework



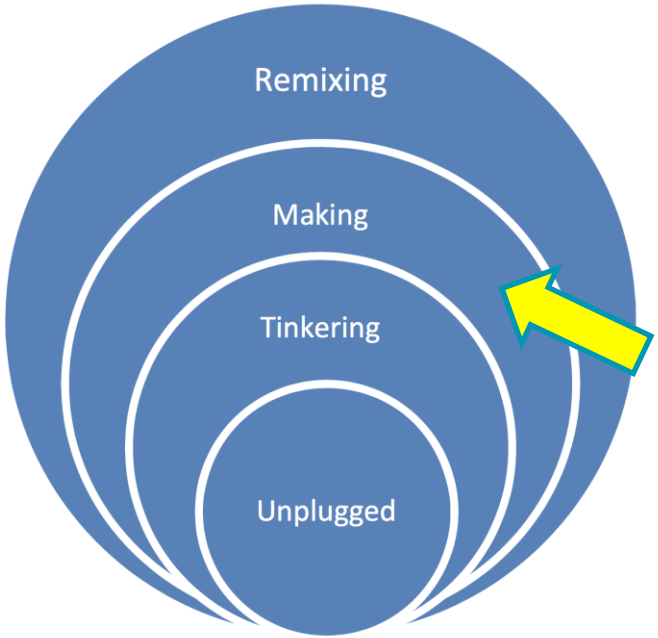
Unplugged experiences focus on CT activities that are implemented without the use of computers. Barriers such as learning a computer programming language or limited access to computers are potentially avoided, particularly for novices and younger students

CT Pedagogical Framework



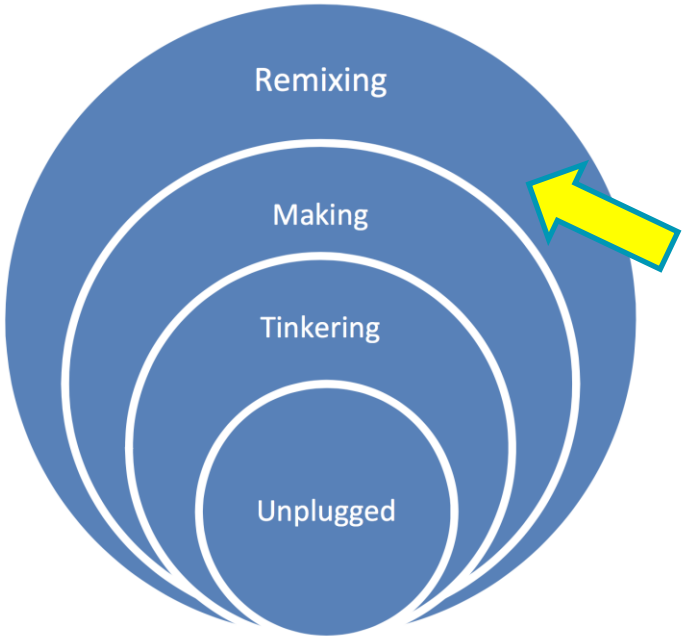
Tinkering experiences primarily involve taking things apart and engaging in changes and/or modifications to existing objects. These objects can be building blocks, puzzles, digital or electronic simulations, programming code, and so forth. During tinkering, students are not constructing an object, digital or otherwise, but rather exploring changes to existing objects and then considering the implications of the changes.

CT Pedagogical Framework



In making experiences, students are required to problem-solve, make plans, select tools, reflect, communicate, and make connections across concepts. Often making involves practices such as prototyping and testing. The knowledge and understanding that is developed in this these pedagogical experiences could involve the development of foundational skills but for the most part, makes use of foundational skills instead. Students have the potential to learn as they construct and while they share what they do, what they have made, and how they have made it

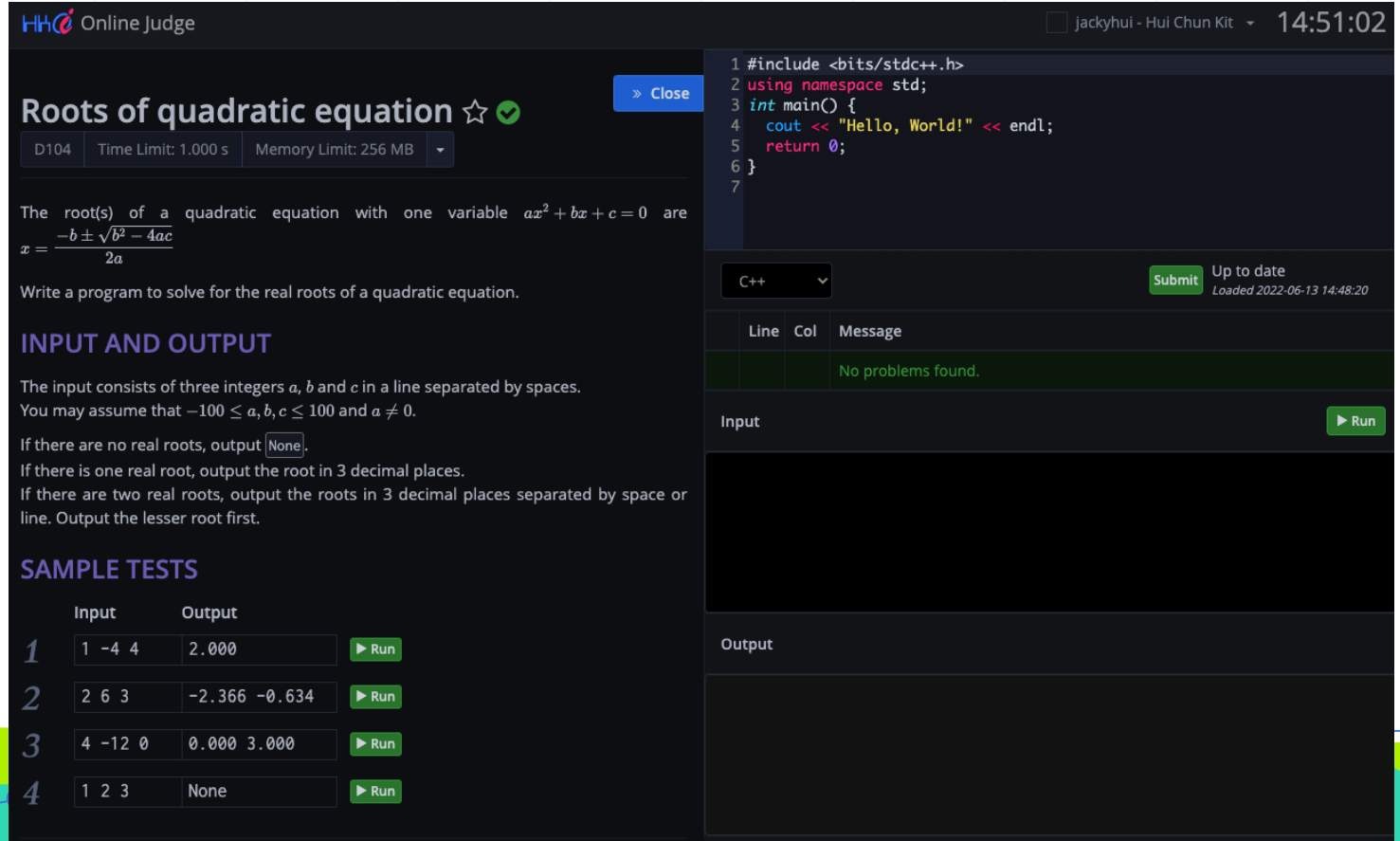
CT Pedagogical Framework



Remixing experiences refer to the appropriation of objects or components of objects for use in other objects or for other purposes. Remixing is sometimes referred to as Bhacking^ or Bdigital sampling.^ Remixing, as used in this context, is more than just sharing an object or sharing and making a minor modification where the objects produced are Blargely derivative, uninteresting, and of poor quality^ (Dasgupta et al. 2016, p. 1439). Remixing experiences involve sharing (intentionally or through Bhacking^) an object and modifying or adapting it in some way and/or embedding it within another object to use it for substantially different purposes.

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education*, *3*(2), 154-171

Handling Learner's diversity



2. John wants to write a program to find the k-th largest element in an array with N elements with index 0 to N-1.

(a) The following is an array with N=7 elements.

Index	0	1	2	3	4	5	6
Value	72	23	67	85	14	2	57

By referring to the array above, what is the k-th largest element when k = 4? _____ (1 mark)

(b) John writes the following subprograms.

`findKthValue1(A, k)`	Return the k-th largest element in array A
`copyArray(A, T)`	Copy the elements from array A to array T
`swap(x, y)`	Swap the contents of variable x and y

(i) Write the pseudocode of `swap(x, y)` by using an extra variable `z`.
(You may assume x and y are passed by reference.)

```
Subprogram swap(x, y)
```

(2 marks)

(ii) Write the pseudocode of `copyArray(A, T)`. You may declare any variable.

```
Subprogram copyArray(A, T)
```

(2 marks)

Supporting students with additional pre-questions

Level 1: Basic Operations with Arrays
- Assign the values 3 to all elements in the array A

Level 2: Identify common patterns
- Write down 1 statement to copy A[0] to T[0]
- Write down 2 more statements to copy A[1] to T[1] and A[2] to T[2]

Level 3: Making a loop

(ii) Write the pseudocode of `copyArray(A, T)`. You may declare any variable.

```
Subprogram copyArray(A, T)
```

(2 marks)

Enriching students with additional post-questions

Modify the program to copy the values from array A to T, with A[0] copied to T[N-1], A[1] to T[N-2] etc

(iii) Complete the following subprogram `findKthValue1(A, k)`.

```
Subprogram findKthValue1(A, k)

Line 1:     copyArray(A, T)
Line 2:     for i from 0 to N-2 do
Line 3:         for j from 0 to [          ] do
Line 4:             if T[j] > T[j+1] then
Line 5:                 swap(T[j], T[j+1])
Line 6:     return [          ]
```

(2 marks)

Some possible Supporting Prompts:

- What should be the data types of each boxes
- Identify familiar algorithm used in this program

(c) John rewrites the algorithm and defines a new subprogram.

findKthValue2(A, k)	Return the k-th largest element in array A

(i) Complete the following subprogram `findKthValue2(A, k)`.

```
Subprogram findKthValue2(A, k)

Line 1:     copyArray(A, T)

Line 2:     for i from 0 to N-2 do

Line 3:         maxIndex ← i

Line 4:         for j from i+1 to N-1 do

Line 5:             if [          ] > T[maxIndex]    then

Line 6:                [          ]

Line 7:                [          ]

Line 8:     return T[k-1]
```

(3 marks)

Some possible Supporting Prompts:

- What should be the data types of each boxes
- Identify familiar algorithm used in this program

(c) John rewrites the algorithm and defines a new subprogram.

findKthValue2(A, k)	Return the k-th largest element in array A

(i) Complete the following subprogram findKthValue2(A, k).

```
Subprogram findKthValue2(A, k)

Line 1:     copyArray(A, T)

Line 2:     for i from 0 to N-2 do

Line 3:         maxIndex ← i

Line 4:         for j from i+1 to N-1 do

Line 5:             if [          ] > T[maxIndex]    then

Line 6:                 [                          ]

Line 7:                 [                          ]

Line 8:     return T[k-1]
```

(3 marks)

Enriching students with additional post-questions

Any other possible algorithms to find the k-th largest element?