Programming 2

Tutorial 1

Activity 1

Given these three implementations of the same algorithm (finding the maximum element), identify the common patterns:

C version:

```
int findMax(int arr[], int n) {
    int max = arr[0];
    for (int i = 1; i < n; i++) {
        if (arr[i] > max)
            max = arr[i];
    }
    return max;
}
```

Python version:

```
def find_max(arr):
    max_val = arr[0]
    for num in arr[1:]:
        if num > max_val:
            max_val = num
    return max_val
```

Tasks:

1. Write the Java version
2. List 5 similarities across all three versions
3. List 3 key differences
4. Explain why the lecture says "once you master one imperative language, it's easy to learn another"

Activity 2

a) C to Java - Bubble Sort

Understand differences in array handling and function declarations

```c
// C Code
#include <stdio.h>

void bubbleSort(int arr[], int n) {
    for (int i = 0; i < n-1; i++) {
        for (int j = 0; j < n-i-1; j++) {
            if (arr[j] > arr[j+1]) {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

int main() {
    int arr[] = {64, 34, 25, 12, 22, 11, 90};
    int n = sizeof(arr)/sizeof(arr[0]);
    bubbleSort(arr, n);
    printArray(arr, n);
    return 0;
}
```

Tasks: Convert this to Java, paying attention to:

- Class structure
- Main method signature
- Array length property
- Output method (System.out.println vs printf)

b) Python to Java - Student Grade Manager

Handle Python's dynamic typing and data structures in Java

```
# Python Code
students = {}

def add_student(name, grade):
    students[name] = grade

def get_average():
    if len(students) == 0:
        return 0
    return sum(students.values()) / len(students)

def get_top_student():
    if not students:
        return None
    return max(students, key=students.get)

# Usage
add_student("Alice", 85)
add_student("Bob", 92)
add_student("Charlie", 78)

print(f"Average grade: {get_average()}")
print(f"Top student: {get_top_student()}")
```

Task: Convert to Java considering:

- Static typing (int, String, double)
- HashMap usage
- Method return types
- Null handling

c) JavaScript to Java - Simple Calculator

Convert from dynamically-typed to statically-typed OOP

```
// JavaScript Code
class Calculator {
    constructor() {
        this.result = 0;
    }

    add(a, b) {
        this.result = a + b;
        return this.result;
    }

    subtract(a, b) {
        this.result = a - b;
        return this.result;
    }

    multiply(a, b) {
        this.result = a * b;
        return this.result;
    }

    divide(a, b) {
        if (b === 0) {
            console.log("Cannot divide by zero");
            return null;
        }
        this.result = a / b;
        return this.result;
    }
}

const calc = new Calculator();
console.log(calc.add(5, 3));
console.log(calc.multiply(4, 2));
```

Task: Convert to Java, addressing:

- Type declarations
- Exception handling for division by zero
- Constructor syntax
- Access modifiers

Activity 3

Write a program that will print your initials to standard output in letters that are nine lines tall. Each big letter should be made up of a bunch of *'s. For example, if your initials were "DJE", then the output would look something like:

```
******          *************    **********
**    **              **         **
**     **             **         **
**       **           **         **
**       **           **         ********
**       **     **    **         **
**     **        **   **         **
**    **           ** **         **
*****               ****         **********
```

Activity 4

Write a program that asks the user's name, and then greets the user by name. Before outputting the user's name, convert it to upper case letters. For example, if the user's name is Fred, then the program should respond "Hello, FRED, nice to meet you!".

Activity 5

If you have `N` eggs, then you have `N/12` dozen eggs, with `N%12` eggs left over (this is essentially the definition of the `/` and `%` operators for integers). Write a program that asks the user how many eggs she has and then tells the user how many dozen eggs she has and how many extra eggs are left over.

A gross of eggs is equal to `144` eggs. Extend your program so that it will tell the user how many gross, how many dozen, and how many left over eggs she has.

For example, if the user says that she has `1342` eggs, then your program would respond with: Your number of eggs is `9` gross, `3` dozen, and `10` since `1342` is equal to `9*144 + 3*12 + 10`.

Activity 6

```
/**
* This class implements a simple program that
* will compute the amount of interest that is
* earned on 17,000 invested at an interest
* rate of 0.07 for one year. The interest and
* the value of the investment after one year are
* printed to standard output.
*/
```

```
9  public class Interest {
10 public static void main(String[] args) {
11 /* Declare the variables. */
12 double principal;
13 double rate;
14 double interest;
15 /* Do the computations. */
16 principal = 17000;
17 rate = 0.07;
18 interest = principal * rate;
19 principal = principal + interest;
20 /* Output the results. */
21 System.out.print("The interest earned is ");
22 System.out.println(interest);
23 System.out.print("The value of the investment after one year is ");
24 System.out.println(principal);
25 } // end of main()
26 } // end of class Interest
```

Answer the following questions about program `Interest`:

a) Change the statement at line 16 to the following statement:

```
principal = 17000.0;
```

Compile and run the program again. Does it work? Why do you think that is?

b) Change the statement at line 12 to the following statement:

```
int principal;
```

Compile the program again. Does it work? Why do you think that is? Can you fix it without reversing the change?

Activity 7 (optional)

There is a shop that has:

`N` types of products. For each integer `i` from `1` to `N`, the i-th type of product has a price of P_i Dong each.

We decide to purchase `M` type of products from `1` to `M`, and we will buy Q_i each type.

We will pay the total price of the products purchased.

Calculate the amount we will pay.

All input values are integers and you will read it from file `input.txt`

Content of `input.txt`:

```
N M
P1 Q1
P2 Q2
….
PM QM
….
PN
```

Example content of file `input.txt`:

```
7 3
500 2
600 3
700 1
300
1000
1100
100
```

Submission

Submit a **zip** file containing all Java programs to this tutorial's submission box in the course website on FIT LMS.