UNIVERSITI MALAYA
UNIVERSITI MALAYA

PEPERIKSAAN IJAZAH SARJANA MUDA SAINS KOMPUTER
EXAMINATION FOR THE DEGREE OF BACHELOR OF COMPUTER SCIENCE

SESI AKADEMIK 2023/2024 : SEMESTER II
ACADEMIC SESSION 2023/2024 : SEMESTER II

WIX1002 : Asas-Asas Pengaturcaraan
Fundamentals of Programming

Jun/Julai 2024
June/July 2024

Masa: 3 jam 30 minit
Time: 3 Hour 30 minute

ARAHAN KEPADA CALON:
INSTRUCTIONS TO CANDIDATES:

Calon dikehendaki menjawab **SEMUA** soalan.
*Candidates should answer **ALL** questions.*

(Kertas soalan ini mengandungi 2 soalan dalam 4 halaman yang dicetak)
(This question paper consists of 2 questions on 4 printed pages)

1. Tulis program yang mengambil tatasusunan 2-dimensi yang mewakili papan *tic-tac-toe* dan tentukan pemenang atau jika permainan itu seri. Fail input merupakan beberapa grid 3x3 yang mewakili papan *tic-tac-toe*, dengan setiap sel mengandungi sama ada 'X', 'O' atau '. ' (noktah). Papan dipisahkan dengan '#'.

   Program anda harus memaparkan 'X' jika pemain X telah menang, 'O' jika pemain O telah menang, "*Draw*" jika permainan seri, dan " *None (on going)*" jika permainan masih berlangsung.

   Write a program that takes a 2-dimensional array representing a tic-tac-toe board and determines the winner or if the game is a draw. The input file will be multiple 3x3 grid representing the tic-tac-toe board, with each cell containing either 'X', 'O', or '. ' (period). Boards are separated by a '#'.

   Your program should output 'X' if player X has won, 'O' if player O has won, "Draw" if the game is a draw, and "None (on going)" if the game is still ongoing.

Contoh input:
Sample input:

```
XOX
OOX
XXO
#
X.X
OOO
X..
#
XOX
OXO
OXX
#
X..
O..
...
#
```

Contoh output:
Sample output:

```
Game 1:
Tic-Tac-Toe Board:
X O X
O O X
X X O
Winner: Draw

Game 2:
Tic-Tac-Toe Board:
X   X
O O O
X
Winner: O

Game 3:
Tic-Tac-Toe Board:
X O X
O X O
O X X
Winner: X

Game 4:
Tic-Tac-Toe Board:
X
O

Winner: None (on going)
```

(25 markah/*marks*)

2. Anda diberikan direktori yang mengandungi beberapa fail, setiap satu mewakili satu kerja penghitungan. Setiap fail kerja mengandungi maklumat tentang satu kerja, termasuk nama kerja, *partition*, memori dan *Quality of Service (QoS)*.

You are given a directory containing several files, each representing a job. Each job file consists of information about a single job, including the job name, partition, memory, and Quality of Service (QoS).

Format untuk setiap fail kerja adalah seperti berikut:
The format of each job file is as follows:

```
JobName: JobName1
Partition: Partition1
Memory: Memory1
QoS: QoS1
```

Tulis satu program Java ***JobReader*** untuk membaca semua fail kerja dari direktori dan mencetak butiran setiap kerja. Tambahkan satu metod untuk menunjukkan laporan statistik kerja yang ringkas, contohnya berapa banyak kerja setiap partition, berapa banyak kerja untuk setiap QoS.

Tentukan kelas ***Job*** untuk mewakili setiap kerja, dengan atribut berikut:
- ***jobName*** (String): Nama kerja.
- ***partition*** (String): Pembahagian kerja [*cpu-epyc-genoa, gpu-a100* dan *gpu-v100s*].
- ***memory*** (String): Memori yang diperlukan untuk kerja.
- ***qos*** (String): Kualiti Perkhidmatan untuk kerja [*normal, long* dan *debug*].

*Write a Java program **JobReader** to read all job files from the directory and print the details of each job. Add a method to show simple job statistic report, for instance how many jobs per partition, how many jobs for each QoS.*

*Define a **Job** class to represent each job, with the following attributes:*
- ***jobName** (String): The name of the job.*
- ***partition** (String): The partition of the job [cpu-epyc-genoa, gpu-a100 and gpu-v100s].*
- ***memory** (String): The memory required for the job.*
- ***qos** (String): The Quality of Service of the job [normal, long and debug].*

Metod **main** untuk **JobReader** adalah seperti yang diberikan di bawah:
*The **main** method of the **JobReader** is given as below:*

```
public class JobReader {

   public static void main(String[] args) {
      Job[] jobs = readJobsFromDirectory("directory_path");
      // Change "directory_path" to your directory path

      listAllJobs(jobs);
      generateStatistics(jobs);
   }
}
```

Contoh output:
Sample output:

```
List of all jobs:
Job Name: Job2, Partition: gpu-a100, Memory: 8GB, QoS: long
Job Name: Job4, Partition: cpu-epyc-genoa, Memory: 8GB, QoS: long
Job Name: Job3, Partition: gpu-v100s, Memory: 16GB, QoS: debug
Job Name: Job1, Partition: cpu-epyc-genoa, Memory: 4GB, QoS: normal

Partition Statistics:
gpu-a100: 1 jobs
gpu-v100s: 1 jobs
cpu-epyc-genoa: 2 jobs

QoS Statistics:
normal: 1 jobs
debug: 1 jobs
long: 2 jobs
```

(25 markah/*marks*)

TAMAT
END