UNIVERSITI MALAYA
UNIVERSITY OF MALAYA

PEPERIKSAAN IJAZAH SARJANA MUDA SAINS KOMPUTER / SARJANA MUDA TEKNOLOGI MAKLUMAT
EXAMINATION FOR THE DEGREE OF BACHELOR OF COMPUTER SCIENCE / BACHELOR OF INFORMATION TECHNOLOGY

SESI AKADEMIK 2019/2020 : SEMESTER I
ACADEMIC SESSION 2019/2020 : SEMESTER I

WIX1002 : Asas-Asas Pengaturcaraan
Fundamentals of Programming

Dis 2019/Jan 2020
Dec 2019/Jan 2020

Masa: 3 jam 30 minit
Time: 3 hours 30 minutes

ARAHAN KEPADA CALON:
INSTRUCTIONS TO CANDIDATES:

Calon dikehendaki menjawab **SEMUA** soalan (50 markah).
*Answer **ALL** questions (50 marks).*

(Kertas soalan ini mengandungi 5 soalan dalam 12 halaman yang dicetak)
(This question paper consists of 5 questions on 12 printed pages)

1. Aturcara dalam fail **Q1.java** mengandungi banyak ralat. Betulkan kesemua ralat tersebut. (Salin fail **Q1.java** dari direktori akaun peperiksaan anda. Selepas aturcara dibetulkan, namakan fail tersebut sebagai [**matricNumberQ1.java**; contoh: **WIA190000Q1.java**] dan salin fail ini ke direktori akaun peperiksaan.)

*The program in the **Q1.java** file contains many errors. Correct all errors. (Copy the **Q1.java** file from your exam account directory. After the program has been corrected, name the file as [**matricNumberQ1.java**; example: **WIA190000Q1.java**] and copy this file to your exam account directory.)*

```
import java.util.Scanner;
// Filename: Q1.java
public class Q1 {
   public static void main(String[] args) {
      System.out.println("This program changes all the odd numbers in the array into even
numbers.");
      Scanner s = new Scanner(System.out);
      int size=5;
      int num = new int[size];
      System.out.print("Enter five integer numbers : ");
      for (int i=1; i<size; i++) {
        num[i] = s.nextDouble();
      }
      convert(num[0]);
      System.out.print("The numbers are : ");
      for (int i=0; i<size; i++) {
        System.out.print(num + " ");
      }
      System.out.println();
   }

   public void convert(int[] a) {
      for (int i=0; i<a.length(); i++) {
         if (a%2==1)
            a[i]+=2;
      }
   }
}
```

Contoh *output*:
Sample output:

```
This program changes all the odd numbers in the array into even numbers.
Enter five integer numbers : 21 14 36 17 9
The numbers are : 22 14 36 18 10
```

(5 markah/*marks*)

2. Sistem koordinat geografi digunakan untuk menentukan lokasi di bumi dengan koordinat longitud dan latitud. Tulis satu aturcara Java yang mengira jarak di antara dua koordinat. Anda boleh menggunakan perpustakaan Java **Math**. Aturcara ini mesti mempunyai kaedah-kaedah berikut:

   - Satu kaedah untuk meminta pengguna memasukkan arah (N, S, E, W), darjah, minit dan saat.
   - Satu kaedah untuk menukar input ke darjah desimal berdasarkan formula dan jadual di bawah.

   A geographic coordinate system is used to determine locations on the earth with longitude and latitude coordinates. Write a Java program that calculates the distance between two coordinates. You can use the Java ***Math*** *library. The program must consist of the following methods:*

   - *A method to request the user to enter the direction (N, S, E, W), degree, minute and second.*
   - *A method to convert the input into decimal degree based on the formula and table below.*

   *decimal degree = (Sign) degree + (minute * 60 + second) / 3600 (Type)*

Direction	Sign	Type
N	+	Latitude
S	-	Latitude
E	+	Longitude
W	-	Longitude

   - Satu kaedah untuk mengira jarak menggunakan formula *haversine* seperti di bawah.

   A method to calculate the distance using haversine formula as below.

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \times \cos\varphi_2 \times \sin^2(\Delta\lambda/2)$$
$$c = 2 \times \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$$
$$d = R \times c$$

   φ *is latitude,* $\Delta\varphi$ is the difference between two latitudes
   λ *is longitude,* $\Delta\lambda$ is the difference between two longitudes
   (All angles need to convert to radians)
   R *is earth's radius (6371), d is the distance.*

(Simpan aturcara tersebut dalam fail **Main.java**. Salin fail ini ke direktori akaun peperiksaan anda dan namakan semula sebagai [**matricNumberQ2.java**; contoh: **WIA190000Q2.java**]).
*(Save the program in the **Main.java** file. Copy this file to your exam account directory and rename as [**matricNumberQ2.java**; example: **WIA190000Q2.java**]).*

Contoh *output*:
Sample output:

```
Enter Location 1
Enter Direction: [N, S, E, W]: N
Enter Degree, Minute and Second : 32 45 51
Enter Direction: [N, S, E, W]: W
Enter Degree, Minute and Second : 21 19 8
Enter Location 2
Enter Direction: [N, S, E, W]: S
Enter Degree, Minute and Second : 13 22 58
Enter Direction: [N, S, E, W]: E
Enter Degree, Minute and Second : 8 11 20
Location 1 : 32.764167 Latitude , -21.318889 Longitude
Location 2 : -13.382778 Latitude , 8.188889 Longitude
Distance : 6013.08 KM
```

(10 markah/*marks*)

3. Ahmad menjalankan eksperimen dan menyimpan data dalam fail teks bernama **raw.txt**. Fail tersebut mengandungi nombor-nombor genap dari 2-10. Tuliskan satu aturcara untuk membina jadual pengagihan frekuensi. Jadual pengagihan frekuensi meringkaskan nilai dan kekerapannya. Kemudian, cari mod data. (Nombor yang paling kerap)

*Ahmad conducted an experiment and save the data in a text file named **raw.txt**. The text file consists of even number from 2-10. Write a Java program to create a frequency distribution table. A frequency distribution table summarizes values and their frequency. Then, find the mode of the data. (The number that occurs most frequently)*

(Simpan aturcara tersebut dalam fail **Main.java**. Salin fail ini ke direktori akaun peperiksaan anda dan namakan semula sebagai [**matricNumberQ3.java**; contoh: **WIA190000Q3.java**]).
*(Save the program in the **Main.java** file. Copy this file to your exam account directory and rename as [**matricNumberQ3.java**; example: **WIA190000Q3.java**]).*

Contoh *output*:
Sample output:

```
Frequency Distribution Table
2 : 9
4 : 9
6 : 13
8 : 10
10 : 9
The mode of the dataset is : 6
```

(10 markah/*marks*)

4. *High Performance Computing* (HPC) kluster mengumpulkan kuasa penghitungan banyak komputer individu yang dikenali sebagai nod perhitungan, untuk menghasilkan prestasi yang lebih tinggi daripada yang dapat keluar dari komputer meja biasa atau stesen kerja untuk menyelesaikan masalah besar dalam sains, kejuruteraan, atau perniagaan. Nod perhitungan disambungkan dengan rangkaian berkelajuan tinggi dan penjadual digunakan untuk menguruskan kerja yang dijalankan di seluruh komputer teragih yang ketat diganding. *UM Data Intensive Computing Centre* (DICC) mengoperasikan 700 CPU HPC kluster dengan menggunakan TORQUE sebagai pengurus sumber teragih. Fail log di bawah adalah sampel kecil fail log penjadual yang menjejaki aktiviti harian.

High Performance Computing (HPC) cluster aggregating computing power of many individual computers known as compute nodes, to deliver much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business. The compute nodes are connected with high-speed network and a scheduler is used to manage the jobs running across these tightly-coupled distributed computers. UM Data Intensive Computing Centre (DICC) is operating a 700 CPUs HPC cluster using TORQUE as the distributed resource manager. The log file below is a small sample of the scheduler log file that keep track of the daily activities.

```
03/01/2017 00:44:56;Q;14126.ce2.dicc.um.edu.my;queue=um
03/01/2017 00:45:00;S;14126.ce2.dicc.um.edu.my;user=weng group=um jobname=Gaussian queue=um
ctime=1488329096 qtime=1488329096 etime=1488329096 start=1488329100
owner=weng@ce2.dicc.um.edu.my
exec_host=compute07.sifir.um.edu.my/63+compute07.sifir.um.edu.my/62+compute07.sifir.um.edu.my
/61+compute07.sifir.um.edu.my/60+compute07.sifir.um.edu.my/59+compute07.sifir.um.edu.my/58+co
mpute07.sifir.um.edu.my/57+compute07.sifir.um.edu.my/56+compute07.sifir.um.edu.my/55+compute0
7.sifir.um.edu.my/54+compute07.sifir.um.edu.my/53+compute07.sifir.um.edu.my/52+compute07.sifi
r.um.edu.my/51+compute07.sifir.um.edu.my/50+compute07.sifir.um.edu.my/49+compute07.sifir.um.e
du.my/48 Resource_List.nednodes=1:ppn=16 Resource_List.nodect=1 Resource_List.nodes=1:ppn=16
Resource_List.walltime=700:00:00
03/01/2017 03:20:22;Q;14127.ce2.dicc.um.edu.my;queue=um
03/01/2017 03:20:23;S;14127.ce2.dicc.um.edu.my;user=wiliam group=um jobname=Amber-CPU-
mmpbsaden4 queue=um ctime=1488338422 qtime=1488338422 etime=1488338422 start=1488338423
owner=wiliam@ce2.dicc.um.edu.my
exec_host=masternode.sifir.um.edu.my/15+masternode.sifir.um.edu.my/14+masternode.sifir.um.edu
.my/13+masternode.sifir.um.edu.my/12+masternode.sifir.um.edu.my/11+masternode.sifir.um.eduzmy
/10+masternode.sifir.um.edu.my/9+masternode.sifir.um.edu.my/8+masternode.sifir.um.edu.my/7+ma
sternode.sifir.um.edu.my/6+masternode.sifir.um.edu.my/5+masternode.sifir.um.edu.my/4+masterno
de.sifir.um.edu.my/3+masternode.sifir.um.edu.my/2+masternode.sifir.um.edu.my/1+masternode.sif
```

```
ir.um.edu.my/0 Resource_List.neednodes=1:ppn=16 Resource_List.nodect=1
Resource_List.nodes=1:ppn=16 Resource_List.walltime=700:00:00
03/01/2017 03:53:09;Q;14129.ce2.dicc.um.edu.my;queue=um
03/01/2017 03:53:11;S;14129.ce2.dicc.um.edu.my;user=sharif group=um jobname=Insrea queue=um
ctime=1488340389 qtime=1488340389 etime=1488340389 start=1488340391
owner=sharif@ce2.dicc.um.edu.my
exec_host=masternode.sifir.um.edu.my/7+masternode.sifir.um.edu.my/6+masternode.sifir.um.edu.m
y/5+masternode.sifir.um.edu.my/4+masternode.sifir.um.edu.my/3+masternode.sifir.um.edu.my/2+ma
sternode.sifir.um.edu.my/1+masternode.sifir.um.edu.my/0 Resource_List.neednodes=1:ppn=8
Resource_List.nodect=1 Resource_List.nodes=1:ppn=8 Resource_List.walltime=700:00:00
03/01/2017 03:53:11;E;14129.ce2.dicc.um.edu.my;user=sharif group=um jobname=Insrea queue=um
ctime=1488340389 qtime=1488340389 etime=1488340389 start=1488340391
owner=sharif@ce2.dicc.um.edu.my
exec_host=masternode.sifir.um.edu.my/7+masternode.sifir.um.edu.my/6+masternode.sifir.um.edu.m
y/5+masternode.sifir.um.edu.my/4+masternode.sifir.um.edu.my/3+masternode.sifir.um.edu.my/2+ma
sternode.sifir.um.edu.my/1+masternode.sifir.um.edu.my/0 Resource_List.neednodes=1:ppn=8
Resource_List.nodect=1 Resource_List.nodes=1:ppn=8 Resource_List.walltime=700:00:00
session=9647 end=1488340391 Exit_status=134 resources_used.cput=00:00:00
resources_used.mem=0kb resources_used.vmem=0kb resources_used.walltime=00:00:00
03/01/2017 04:16:30;E;14127.ce2.dicc.um.edu.my;user=wiliam group=um jobname=Amber-CPU-
mmpbsaden4 queue=um ctime=1488338422 qtime=1488338422 etime=1488338422 start=1488338423
owner=wiliam@ce2.dicc.um.edu.my
exec_host=masternode.sifir.um.edu.my/15+masternode.sifir.um.edu.my/14+masternode.sifir.um.edu
.my/13+masternode.sifir.um.edu.my/12+masternode.sifir.um.edu.my/11+masternode.sifir.um.edu.my
/10+masternode.sifir.um.edu.my/9+masternode.sifir.um.edu.my/8+masternode.sifir.um.edu.my/7+ma
sternode.sifir.um.edu.my/6+masternode.sifir.um.edu.my/5+masternode.sifir.um.edu.my/4+masterno
de.sifir.um.edu.my/3+masternode.sifir.um.edu.my/2+masternode.sifir.um.edu.my/1+masternode.sif
ir.um.edu.my/0 Resource_List.neednodes=1:ppn=16 Resource_List.nodect=1
Resource_List.nodes=1:ppn=16 Resource_List.walltime=700:00:00 session=8071 end=1488341790
Exit_status=1 resources_used.cput=12:03:36 resources_used.mem=17793264kb
resources_used.vmem=17793264kb resources_used.walltime=00:56:07
03/01/2017 04:16:40;Q;14130.ce2.dicc.um.edu.my;queue=um
03/01/2017 04:16:40;S;14130.ce2.dicc.um.edu.my;user=wiliam group=um jobname=Amber-CPU-
mmpbsaden4 queue=um ctime=1488341800 time=1488341800 etime=1488341800 start=1488341800
owner=wiliam@ce2.dicc.um.edu.my
exec_host=masternode.sifir.um.edu.my/15+masternode.sifir.um.edu.my/14+masternode.sifir.um.edu
.my/13+masternode.sifir.um.edu.my/12+masternode.sifir.um.edu.my/11+masternode.sifir.um.edu.my
/10+masternode.sifir.um.edu.my/9+masternode.sifir.um.edu.my/8+masternode.sifir.um.edu.my/7+ma
sternode.sifir.um.edu.my/6+masternode.sifir.um.edu.my/5+masternode.sifir.um.edu.my/4+masterno
de.sifir.um.edu.my/3+masternode.sifir.um.edu.my/2+masternode.sifir.um.edu.my/1+masternode.sif
ir.um.edu.my/0 Resource_List.neednodes=1:ppn=16 Resource_List.nodect=1
Resource_List.nodes=1:ppn=16 Resource_List.walltime=700:00:00
03/01/2017 08:54:50;Q;14132.ce2.dicc.um.edu.my;queue=gpu
03/01/2017 08:54:52;S;14132.ce2.dicc.um.edu.my;user=afandi group=gpu jobname=Amber-GPU1
queue=gpu ctime=1488358490 qtime=1488358490 etime=1488358490 start=1488358492
owner=afandi@ce2.dicc.um.edu.my exec_host=gpuserver02.dicc.um.edu.my/0
Resource_List.neednodes=gpuserver02.dicc.um.edu.my:ppn=1 Resource_List.nodect=1
Resource_List.nodes=gpuserver02.dicc.um.edu.my:ppn=1 Resource_List.walltime=700:00:00
03/01/2017 08:54:52;E;14132.ce2.dicc.um.edu.my;user=afandi group=gpu jobname=Amber-GPU1
queue=gpu ctime=1488358490 qtime=1488358490 etime=1488358490 start=1488358492
owner=afandi@ce2.dicc.um.edu.my exec_host=gpuserver02.dicc.um.edu.my/0
Resource_List.neednodes=gpuserver02.dicc.um.edu.my:ppn=1 Resource_List.nodect=1
Resource_List.nodes=gpuserver02.dicc.um.edu.my:ppn=1 Resource_List.walltime=700:00:00
session=0 end=1488358492 Exit_status=-2 resources_used.cput=00:00:00 resources_used.mem=0kb
resources_used.vmem=0kb resources_used.walltime=00:00:00
```

Rekod, iaitu log aktiviti, bermula dengan tarikh aktiviti direkodkan diikuti oleh atribut yang lain daripada pelaksanaan masing-masing. Terdapat tiga jenis rekod dalam fail log ini: Q, menunjukkan kerja telah dihantarkan kepada penjadual, S, percubaan untuk memulakan kerja yang telah dibuat, dan E, kerja itu telah habis (sama ada dilaksanakan dengan jayanya atau diakhiri dengan ralat). Sebagai contoh, rekod di bawah menunjukkan bahawa kerja dengan ID **14126**, yang dihantar oleh pengguna **weng**, telah mula dilaksanakan pada **1488329100**, yang merupakan *timestamp* Linux yang menunjukkan bilangan saat yang telah berlalu sejak 1 Januari 1970 (tengah malam UTC / GMT) .

*The record, i.e. the activity log, starts with the date when the activity been recorded followed by other attributes of the respective execution. There are three types of record in this log file: Q, indicates a job has been submitted to the scheduler, S, an attempt to start the job has been made, and E, the job has exited (either run successfully or terminated with error). For instance, the record below shows that the job with job ID **14126**, submitted by user **weng**, has been started at **1488329100**, which is a Linux timestamp indicating the number of seconds that have elapsed since January 1, 1970 (midnight UTC/GMT).*

```
03/01/2017 00:45:00;S;14126.ce2.dicc.um.edu.my;user=weng group=um jobname=Gaussian queue=um
ctime=1488329096 qtime=1488329096 etime=1488329096 start=1488329100
owner=weng@ce2.dicc.um.edu.my
```

Rekod jenis Q hanya mempunyai 2 atribut, manakala rekod jenis S, seperti yang ditunjukkan di atas, mempunyai 15 atribut dan rekod jenis E mempunyai 22 atribut. Atribut dipisahkan dengan ruang dan setiap rekod diakhiri dengan barisan baru.
Q type records have only 2 attributes, while S type records, as shown above, have 15 attributes and E type records have 22 attributes. The attributes are separated with a space and each record is ended with a new line.

Tulis program analisis log yang mudah untuk membantu pakar HPC DICC untuk memantau prestasi HPC *cluster*. Program anda harus melaksanakan tiga fungsi:

a) Baca fail log bernama ***sampleLog*** dan mencetak rekod yang dibaca dari fail log dalam format yang diberikan, dengan jumlah rekod yang dibaca.
Petunjuk: gunakan tatasusunan dua dimensi untuk menyimpan rekod, di mana setiap baris mewakili rekod dan lajur adalah atributnya.

b) Cetak bilangan jumlah kerja yang diserahkan oleh setiap pengguna.
Petunjuk: baca rekod S dan baca elemen terakhir dari atribut kedua.

c) Cetak status pelaksanaan bagi setiap kerja.
Petunjuk: ID kerja ialah elemen ketiga dari atribut kedua rekod S, dan berakhir dengan `.ce.dicc.um.edu.my`. Anda boleh menggunakan metod String.split untuk mengekstraknya, tetapi ingat bahawa titik itu bukan tanda hujung biasa dan anda harus menggunakan aksara lepasan "\\.". Jika status tidak dibaca dari fail log, biarkan dengan nilai *null*. Sekiranya kerja itu gagal dilaksanakan, cetak *exit status* sebagai mesej ralat, bukan masa tamat. Kedua-dua *exit status* dan masa tamat boleh didapati dalam rekod E.

Write a simple log analysis program to help the HPC specialist of DICC to monitor the performance of the HPC cluster. Your program should perform three functions:

*a) Read the log file named **sampleLog** and print the records read from the log file in the given format, with the total number of records read.*
Hint: use a two-dimensional array to store the record, where each row represents a record and the columns are the its attributes.

b) Print the total number of jobs submitted by each user.
Hint: read only the type S record and read the last element of the second attribute.

c) *Print the execution status of each and every job.*
Hint: the job ID is the third element of the second attribute of the S type record, and ended with `.ce.dicc.um.edu.my`. *You can use String.split method to extract it, but remember that period is a not a common delimiter and you should use escape character "\\.". If the status is not read from the log file, just leave it as it is with a null value. If the job did not successfully been executed, print the exit status as error message, instead of the end time. Both exit status and end time can be found in the E type record.*

(Simpan aturcara tersebut dalam fail **Main.java**. Salin fail ini ke direktori akaun peperiksaan anda dan namakan semula sebagai [**matricNumberQ4.java**; contoh: **WIA190000Q4.java**]).
*(Save the program in the **Main.java** file. Copy this file to your exam account directory and rename as [**matricNumberQ4.java**; example: **WIA190000Q4.java**]).*
Contoh output: [rekod 4 hingga sebahagian besar rekod 13 telah dikeluarkan]
Sample output: [record 4 to big part of record 13 have been cut out]

```
Reading from log file...
****************************************
Records read from log file

Record 1:
    03/01/2017
    00:44:56;Q;14126.ce2.dicc.um.edu.my;queue=um

Record 2:
    03/01/2017
    00:45:00;S;14126.ce2.dicc.um.edu.my;user=weng
    group=um
    jobname=Gaussian
    queue=um
    ctime=1488329096
    qtime=1488329096
    etime=1488329096
    start=1488329100
    owner=weng@ce2.dicc.um.edu.my
    exec_host=compute07.sifir.um.edu.my/63+compute07.sifir.um.edu.my/62+compute07.sifir.um.ed
u.my/61+compute07.sifir.um.edu.my/60+compute07.sifir.um.edu.my/59+compute07.sifir.um.edu.my/5
8+compute07.sifir.um.edu.my/57+compute07.sifir.um.edu.my/56+compute07.sifir.um.edu.my/55+comp
ute07.sifir.um.edu.my/54+compute07.sifir.um.edu.my/53+compute07.sifir.um.edu.my/52+compute07.
sifir.um.edu.my/51+compute07.sifir.um.edu.my/50+compute07.sifir.um.edu.my/49+compute07.sifir.
um.edu.my/48
    Resource_List.neednodes=1:ppn=16
    Resource_List.nodect=1
    Resource_List.nodes=1:ppn=16
    Resource_List.walltime=700:00:00

Record 3:
    03/01/2017
    03:20:22;Q;14127.ce2.dicc.um.edu.my;queue=um

 ...

    resources_used.cput=00:00:00
    resources_used.mem=0kb
    resources_used.vmem=0kb
    resources_used.walltime=00:00:00

total records read: 13

****************************************
Print user submission stat

User                Jobs Submitted
----                --------------
```

```
weng                1
wiliam              2
sharif              1
afandi              1

****************************************
Print jobs status

Job ID      Submitted (queue)      Started (start time)   Exited (end time/error)
------      -----------------      --------------------   -----------------------
14126       Y (um)                 Y (1488329100)         null
14127       Y (um)                 Y (1488338423)         1488341790
14129       Y (um)                 Y (1488340391)         Error: Exit Status 134
14130       Y (um)                 Y (1488341800)         null
14132       Y (gpu)                Y (1488358492)         Error: Exit Status -2

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

(10 markah/*marks*)

5. Infrastruktur pengkomputeran awan menyediakan antara muka untuk pengguna mengakses sumber virtual (kuasa pengiraan, memori dan penyimpanan). Tulis satu aturcara untuk mewakili infrastruktur pengkomputeran awan. Bina satu kelas **Cloud** yang terdiri daripada ahli berikut:

   - Satu medan untuk pakej awan
   - Satu medan untuk jumlah kos
   - Kaedah *Accessor* untuk jumlah kos
   - Satu pembina yang mengandungi pakej awan
   - Satu kaedah **toString** yang mengembalikan pakej awan dan jumlah kos.

   Cloud computing infrastructure provides an interface for users to access the virtualized resources (computing power, memory and storage). Write a program to represent the Cloud computing infrastructure. Design a ***Cloud*** *class that consists of the following members:*

   - *A field for the Cloud package*
   - *A field for the total cost*
   - *Accessor methods for total cost*
   - *A constructor that contains the Cloud package*
   - *A* ***toString*** *method that returns Cloud package and total cost.*

   (3 markah/*marks*)

   Bina satu kelas **Job**. Kelas ini mempunyai ahli berikut:

   - Satu medan untuk nama kerja
   - Satu medan untuk bilangan tugas
   - Satu medan untuk memori
   - Kaedah *Accessor* untuk semua medan.
   - Satu Pembina yang mengandungi semua medan.
   - Satu kaedah **toString** yang mengembalikan nama kerja, memori dan bilangan tugas.

   Design a ***Job*** *class. The class has the following members:*

- *A field for the name of the job*
- *A field for the number of tasks*
- *A field for the memory*
- *Accessor methods for all the fields*
- *A constructor that contains the above fields.*
- *A **toString** method that returns the name of the job, memory and number of tasks.*

(4 markah/*marks*)

Bina satu kelas **CloudPackage** yang mewarisi kelas Cloud. Kelas ini mempunyai ahli berikut:

- Satu medan untuk jumlah *core*
- Satu medan untuk memori
- Satu medan untuk harga sejam
- Satu pembina yang mengandungi medan-medan di atas.
- Satu kaedah **check** yang mengandungi hujah objek **Job**. Kaedah tersebut memeriksa sama ada kerja boleh dilaksanakan. Kerja boleh dilaksanakan jika pakej awan memenuhi keperluan memori. Memori pakej awan mesti lebih atau sama dengan kerja yang dilaksanakan.
- Satu kaedah **totalCost** yang mengandungi hujah objek **Job**. Kaedah tersebut mengira kos, diberi kerja boleh dilaksanakan selari dalam *multi-cores*. Sebagai contoh, satu kerja yang terdiri daripada 20 tugas dapat dilaksanakan dua kali lebih cepat dalam *dual-core* berbanding dengan *single core*. Semua pakej awan mempunyai kuasa pengkomputeran yang sama dan setiap tugas memerlukan masa 1 jam untuk dilaksanakan.

*Design a **CloudPackage** class that extends the **Cloud** class. The class has the following members:*

- *A field for the numbers of core*
- *A field for the memory*
- *A field for price per hour*
- *A constructor that contains the above field.*
- *A **check** method that consists of an argument of **Job** object. The methods check whether a job can be executed. A job can be executed if the Cloud package fulfills the memory requirement. The Cloud package must have more or equal memory to the job executed.*
- *A **totalCost** method that consists of an argument of **Job** object. The method calculates the cost, given a job can run parallel in multi-cores. Example, a job consists of 20 tasks can run twice faster in dual-core as compared to single core. All the compute instances has the same computing power and each task will require 1 hour to execute.*

(4 markah/*marks*)

Ahmad merancang untuk melaksanakan tiga kerja yang berbeza seperti di Jadual 1 menggunakan pakej awan dalam Jadual 2. Tulis kelas penguji yang membina objek

yang sesuai untuk **Job** dan tatasusunan objek untuk **CloudPackage**. Kemudian, tentukan kos paling rendah yang diperlukan untuk setiap kerja dengan menggunakan pakej awan yang dipilih. Akhirnya paparkan jumlah kos untuk kesemua kerja.

Ahmad is planning to execute three different jobs in Table 1 using the cloud packages in Table 2. Write a tester class that creates the suitable objects for ***Job*** *and array of object for* ***CloudPackage****. Then, determine the least cost used for each task using the chosen Cloud package. Finally display the total cost for all the jobs.*

Table 1 Job

Job Name	***Number of Tasks***	***Memory (GB)***
J1	*252*	*20*
J2	*108*	*10*
J3	*72*	*25*

Table 2 Cloud Package

Package Name	***Number of Cores***	***Memory (GB)***	***Price per hour***
P2-15	*4*	*15*	*1.24*
P2-30	*6*	*30*	*2.11*
R5-20	*4*	*20*	*1.38*
R6-20	*6*	*20*	*1.88*

(4 markah/*marks*)

(Salin fail **Cloud.java**, **ComputeNode.java, Job.java** dan **Main.java** ke dalam direktori akaun peperiksaan anda. Namakan Cloud.java sebagai [**matricNumber Cloud.java**; contoh: **WIA190000Cloud.java**], **ComputeNode.java** sebagai [**matricNumberComputeNode.java**; contoh: **WIA190000ComputeNode.java**], **Job.java** sebagai [**matricNumberJob.java**; contoh: **WIA190000Job.java**] dan **Main.java** sebagai [**matricNumberQ5.java**; contoh: **WIA190000Q5.java**]).

(Copy the ***Cloud.java, ComputeNode.java, Job.java*** *and* ***Main.java*** *files to your exam account directory. Rename the* ***Cloud.java*** *as [****matricNumberCloud.java****; example:* ***WIA190000Cloud.java****], **ComputeNode.java** as [****matricNumberComputeNode.java****; example:* ***WIA190000ComputeNode.java****], **Job.java** as [****matricNumberJob.java****; example:* ***WIA190000Job.java****] and **Main.java** as [****matricNumberQ5.java****; example:* ***WIA190000Q5.java****]).*

Contoh *output*:
Sample output:

```
Job Name : J1 Memory = 20GB Number Of Task = 252
Cloud Package : R6-20  Total Cost= 78.96
Job Name : J2 Memory = 10GB Number Of Task = 108
Cloud Package : P2-15  Total Cost= 33.48
Job Name : J3 Memory = 25GB Number Of Task = 72
Cloud Package : P2-30  Total Cost= 25.32
Total Cost : 137.76
```

TAMAT
END