# encora

# Technical Test

# DEV - E

Date: _____8/4/25_____ City/State: ____Recife/PE____

Course: ____Computer Science____ Educational Institution: ____CESAR School____

Course Duration (in years): __4__ Current Semester: __5__ Graduation Year (expected): __2027__

Availability to work: ☐ 20h ☐ 30h ☒ 40h Estimated Start Date: __effective immediately__

Instructions:

This test consists of 8 multiple choice questions, 1 algorithm implementation and 1 non-technical question. The algorithm is worth 60% of the total score. The non-technical question must be answered in Portuguese.

You may use any blank space on this test as a draft.

Use the table below to record your answers.

Good luck!

### Answer Sheet

|   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| A |   |   |   |   |   |   |   |   |
| B | X |   |   | X |   |   |   |   |
| C |   |   |   |   | X |   |   | X |
| D |   | X | X |   |   | X | X |   |

# Question 1

Given:

```
1.  import java.util.Date;
2.  public class Example {
3.    public static void main(String args[]) {
4.    Date d1 = new Date (92, 11, 31);
5.      Date d2 = new Date (94, 11, 31);
6.      method(d1, d2);
7.    System.out.println("d1=" + d1.getYear() + "\nd2=" + d2.getYear());
8.    }
9.    public static void method(Date d1, Date d2) {
10.     d2.setYear(98);
11.     d1 = d2;
12.   }
13. }
```

What's the output?

A.  d1=92
    d2=94

B.  d1=92
    d2=98

C.  d1=98
    d2=98

D.  d1=98
    d2=94

# Question 2

Given:

```
1.   //*****************************
2.   // file A.java
3.   //*****************************
4.   package a;
5.   public class A {
6.       private int x;
7.       protected int y;
8.       public int m1() {return x;}
9.   }
10.  //*****************************
11.  // file B.java
12.  //*****************************
13.  package b;
14.  import a.A;
15.  public class B extends A {
16.      private int z;
17.      public void m2(A a){
18.          z = x;
19.          z = y;
20.          z = a.m1();
21.      }
22.  }
```

Consider the following statements:
I.    Line 18 is valid because B extends A
II.   Line 19 is not valid because y is protected
III.  Line 20 is valid because m1() is public

A.    Only I and II are correct
B.    Only I and III are correct
C.    Only II and III are correct
D.    Only III is correct

Line 19: z = y;
y is declared as protected, which means:

It is accessible to subclasses, but only when accessed via inheritance, i.e., using this.y.

In this line, we're trying to access y through an object a of type A, not via inheritance.

Since class B is in a different package and y is being accessed through an object, this is not allowed.

Line 20: z = a.m1();
Method m1() is declared as public, meaning it is accessible from anywhere.

So, calling a.m1() is completely valid.

Line 18: z = x;
x is declared as private in class A.

Even though class B extends A, private members are not inherited and are not accessible outside the class where they are defined.

So, line 18 tries to access a private field of another class directly — this is not allowed.

## Question 3

What best defines an abstract class?

A. The class must not have method definitions.
B. The class must have a method definition returning nothing.
C. The class must have a constructor that takes no arguments.
D. The class cannot be instantiated as it is mainly for inheritance.

## Question 4

Consider this small binary tree:

```
        14
       /  \
      2    11
     / \   / \
    1   3 10  30
         /    /
        7    40
```
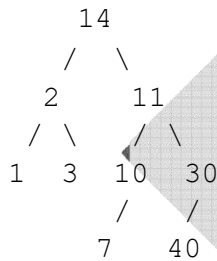
What is the order of nodes visited using a post-order traversal?

A.  1 2 3 7 10 11 14 30 40
B.  1 2 3 14 7 10 11 40 30
C.  1 3 2 7 10 40 30 11 14
D.  14 2 1 3 11 10 7 30 40

## Question 5

Consider the following statements:

I.   A Binary tree is a tree data structure in which each node has at most two child nodes, usually distinguished as "left" and "right", and a tree with $n$ nodes has exactly $n-1$ branches which means its height is always n-1.

II.  A Queue is a FIFO data structure, which means that the first element added to the queue will be the first one to be removed.

III. A Hash Map is a data structure in which, if there's no collision among the keys, you can always find an element in O(1) time, even in the worst case.

A.  Only I and II are correct
B.  Only I and III are correct
C.  Only II and III are correct
D.  I, II and III are correct

Statement I is incorrect because the height of a binary tree is not always n-1; it depends on its structure.

Statement II is correct as queues follow FIFO order.

Statement III is also correct under the assumption of no collisions, which allows O(1) access time.

# Question 6

In the following code, assume that Queue is not thread-safe, there is more than one Producer thread and more than one Consumer thread running and this program is crashing on runtime. In order to fix the code below how should you fill in lines (1), (2), (3) and (4)?

| Global variables |  |
| --- | --- |
| Queue q;<br>(1) | |
| Producer thread | Consumer thread |
| ``` runProducer(){   while(true){     item = new item();     (2)     if (q is not full){       q.enqueue(item);       (3)     }     (4)   } } ``` | ``` runConsumer(){   while(true){     (2)     if (q is not empty){       item = q.dequeue();       (3)     }     (4)   } } ``` |

A. (1) `mutex m;`
   (2) `m.lock();`
   (3)
   (4) `m.unlock();`

B. (1)
   (2)
   (3)
   (4) `if(Consumer) sleep(1); else sleep(2);`

C. (1) `semaphore guard;`
   (2) `wait(guard);`
   (3)
   (4) `signal(guard);`

D. Alternatives A and C are correct.

In this problem, the queue is not thread-safe, and there are multiple producer and consumer threads. This means concurrent access to the queue can lead to race conditions, which explains why the program crashes at runtime. To solve this, the access to the shared queue must be synchronized.

Alternative A uses a mutex. Declaring a mutex variable and locking it before accessing the queue ensures mutual exclusion. Only one thread can access the queue at a time, preventing data races. After the queue operation, the mutex is unlocked. This is a valid and standard approach to synchronization.

Alternative C uses a semaphore. A semaphore can control access to the queue by using wait before the operation and signal afterward. This also guarantees that only one thread interacts with the queue at a time. It is another valid synchronization method.

Alternative B is incorrect because it uses sleep statements instead of proper synchronization. Sleeping does not prevent race conditions and is not a safe way to coordinate access to shared resources.

Therefore, both alternatives A and C are valid solutions, and the correct choice is D.

# Question 7

Considering the following tables and data information, what would be the correct result of the SQL command below?

| Salesperson | | | |
|---|---|---|---|
| ID | Name | Age | Salary |
| 1 | Abe | 61 | 140,000 |
| 2 | Bob | 34 | 44,000 |
| 5 | Chris | 34 | 40,000 |
| 7 | Dan | 41 | 52,000 |
| 8 | Ken | 57 | 115,000 |
| 11 | Joe | 38 | 38,000 |

| Customer | | | |
|---|---|---|---|
| ID | Name | City | Industry_Type |
| 4 | Samsonic | Pleasant | G |
| 6 | Panasung | Oaktown | N |
| 7 | Samony | Jackson | N |
| 9 | Ornange | Hayward | G |
| 8 | Hepoul | Cupertino | I |

| Orders | | | | |
|---|---|---|---|---|
| Number | Order_Date | cust_id | salesperson_id | Amount |
| 10 | 8/2/2010 | 4 | 2 | 540 |
| 20 | 5/6/2012 | 9 | 7 | 150 |
| 30 | 3/12/2012 | 8 | 5 | 1,500 |
| 40 | 1/30/2013 | 4 | 8 | 1,800 |
| 50 | 7/14/2009 | 9 | 1 | 460 |
| 60 | 1/29/2012 | 7 | 2 | 2,400 |
| 70 | 2/3/2012 | 6 | 7 | 600 |
| 80 | 4/1/2013 | 8 | 2 | 2,300 |
| 90 | 3/2/2012 | 6 | 7 | 720 |

```
SELECT Salesperson.Name from Salesperson
WHERE Salesperson.ID NOT IN(
    SELECT Orders.salesperson_id FROM Orders
    INNER JOIN Customer ON Orders.cust_id = Customer.ID
    WHERE Customer.Name = 'Panasung')
AND Salesperson.ID IN
    (SELECT DISTINCT Orders.salesperson_id FROM Orders);
```

A. Bob
   Chris
   Ken

B. Abe
   Bob
   Ken

C. Abe
   Bob
   Chris
   Ken
   Joe

D. Abe
   Bob
   Chris
   Ken

# Question 8

Given this output on a Linux terminal:

```
$ cat linux_distributions.txt
Debian distribution
Ubuntu distribution derived from Debian
Fedora distribution
Red Hat Enterprise Linux distribution derived from Fedora
CentOS distribution derived from Fedora
MINIX and Linux operating system
```

What will be the correct result of the command below?

```
$ cat linux_distributions.txt | grep Fedora | sort
```

A.  Fedora distribution
    Red Hat Enterprise Linux distribution derived from Fedora
    CentOS distribution derived from Fedora

B.  CentOS derived distribution Fedora from
    distribution Fedora
    derived distribution Enterprise Fedora from Hat Linux Red

C.  CentOS distribution derived from Fedora
    Fedora distribution
    Red Hat Enterprise Linux distribution derived from Fedora

D.  distribution Fedora
    derived distribution Enterprise Fedora from Hat Linux Red
    CentOS derived distribution Fedora from

# Subsets

Write the function getSubSets() to compute and return all subsets of a given set A, which has at most 4 elements.

Use the Set data structure to represent all sets and sub-sets of your solution. A Set is a collection that contains no duplicate elements and the order of elements is irrelevant. Consider the following interface defined for Set:

| Method signature | Method description |
|---|---|
| `boolean add(Element e)` | Adds the specified element to this set if it is not already present (optional operation). |
| `boolean addAll(Set s)` | Adds all elements from s that are not already present in this set. |
| `boolean contains(Element e)` | Returns true if this set contains the specified element. |
| `boolean equals(Set s)` | Compares the specified set s with this set for equality. |
| `Iterator<Element> iterator()` | Returns an iterator over the elements in this set. |
| `boolean remove(Element e)` | Removes the specified element from this set if it is present (optional operation). |
| `int size()` | Returns the number of elements in this set (its cardinality). |
| `Element[] toArray()` | Returns an array containing all of the elements in this set. |

*Table: Set interface*

Input example:
```
A=[1,2,3]
```

Output for the given example:
```
[[1,2,3], [1,2], [1,3], [2,3], [1], [2], [3], []]*
```

* this is the content of the Set which should be returned by the function.

Your proposed solution can be written in pseudo-code or any well-known language (C, C++, Java, etc) and you are free to implement any auxiliary functions. Besides, write down a comment to the main function explaining how your function will work like the one below.

```
/**
 * The function below will ...
 * - Obtain the input
 * - Iterate over the elements
 * …
 * - Print the output and return ...
 */
```

# Algorithm Solution

```java
package encora_prova;

import java.util.*;

public class SubsetsGenerator {

    /**
     * The function below will:
     * - Obtain the input set
     * - Convert the set into a list for indexed access
     * - Iterate over all binary combinations from 0 to 2^n - 1
     *   to determine which elements belong to each subset
     * - Build each subset using bitmasking and add it to the result set
     * - Print the result and return all subsets of the input set
     */

    public static Set<Set<Integer>> getSubSets(Set<Integer> input) {
        Set<Set<Integer>> allSubsets = new HashSet<>();
        List<Integer> elements = new ArrayList<>(input);
        int n = elements.size();
        int total = 1 << n; // 2^n subsets

        for (int i = 0; i < total; i++) {
            Set<Integer> subset = new HashSet<>();
            for (int j = 0; j < n; j++) {
                if ((i & (1 << j)) != 0) {
                    subset.add(elements.get(j));
                }
            }
            allSubsets.add(subset);
        }

        return allSubsets;
    }

    public static void main(String[] args) {
        // Example input
        Set<Integer> input = new HashSet<>();
        input.add(1);
        input.add(2);
        input.add(3);

        // Call the function
        Set<Set<Integer>> result = getSubSets(input);

        // Print the subsets
        System.out.println("Subsets of " + input + ":");
        for (Set<Integer> subset : result) {
            System.out.println(subset);
        }
    }
}
```

Qual a disciplina que você mais gostou de cursar na faculdade e por quê? (Responder em português)

A disciplina que eu mais gostei de cursar na faculdade foi Fundamentos de Desenvolvimento de Software (FDS). Achei muito interessante porque pude ter uma visão completa de todas as etapas envolvidas na criação de um software — desde a ideação e prototipação, passando pela definição de MVP, aplicação de metodologias ágeis, até a parte prática de realmente codar e desenvolver.

Foi uma matéria que conectou teoria e prática de forma muito clara, mostrando como funciona o desenvolvimento de um projeto do início ao fim.

Tive a oportunidade de dar deploy no meu primeiro webapp na Cloud da Microsoft, a Azure. Aprendi Django, solidifiquei e aperfeiçoei meus conhecimentos em python e por fim ainda pude trabalhar em equipe com meus amigos e desempenharmos nossas funções de forma clara, ajudando os pontos fracos do outro e trabalhando com nossos pontos fortes para solucionar quaisquer problemas que surgissem.