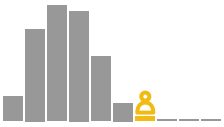
Belguith Nouha (nouha.belguith@expersi.com)

Campagne : Back-end - Expert Domaine(s) : Language independent Langage : Français Date : 18/01/2024

MEILLEUR QUE
>99%
des professionnels



RANG 1 / 1
DURÉE 0h32 / 0h57
SCORE 375 / 600 (62%)

Language independent

375 / 600pts (62%)

MEILLEUR QUE >99% des professionnels

Fiabilité		40 / 155pts
Résolution de problèmes		335 / 445pts

Accéder au rapport détaillé

Question 1: Classification de paquets robotisée

Language independent | 06:46 / 15:00 | 1x (9 sec) | 150 / 150 pts

Question

Objectif
Utiliser le bras robotique de l'usine pour trier les colis.

Règles
Vous travaillez pour une usine autonome. Votre objectif est de trier les colis qui arrivent sur le bon tas en fonction de leur volume et poids. Un colis est encombrant si son volume (Largeur x Hauteur x Profondeur) est supérieur ou égal à 1 000 000 cm3 ou si l'une de ses dimensions est supérieure ou égale à 150 cm. Un colis est lourd si sa masse est supérieure ou égale à 20kg. Vous devez répartir les colis dans 3 tas : STANDARD : les colis normaux (ni encombrants, ni lourds) seront traités normalement. SPECIAL : les colis lourds ou encombrants ne pourront pas être traités automatiquement. REJECTED : les colis à la fois encombrants et lourds seront refusés.

Implémentation
Implémentez la méthode `solve(width, height, length, mass)` (les unités sont le centimètre pour les dimensions et le kilogramme pour la masse). La méthode doit retourner une chaîne de caractères : le nom du tas où placer la boîte.

Conditions de Victoire
Les colis sont répartis sur les bons tas.

Conditions de Défaite
Votre programme indique une action invalide ou fausse.

Contraintes
20 ≤ width, height, length ≤ 200
10 ≤ mass ≤ 1000

Réponse

JAVA

```java
import java.util.*;
import java.io.*;
import java.math.*;

class Player {

    public static String solve(int width, int height, int length, int mass) {
        // Write your code here
        // To debug: System.err.println("Debug messages...");

        if(!isEncombrant(width, height, length, mass) && ! isMass(mass)){
            return "STANDARD";
        }
        else if(isEncombrant(width, height, length, mass) && isMass(mass)){
           return "REJECTED";
        }else {
            return "SPECIAL";
        }
    }

    public static boolean isMass(int mass){
        return mass >= 20;
    }
     public static boolean isEncombrant(int width, int height, int length,int mass){
        return width*height*length >= 1000000 || width >=150 || height >= 150 || length>=
150;
    }
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);

        // game loop
        while (true) {
            int width = in.nextInt();
            int height = in.nextInt();
            int length = in.nextInt();
            int mass = in.nextInt();
            PrintStream outStream = System.out;
            System.setOut(System.err);
            String action = solve(width, height, length, mass);
            System.setOut(outStream);
            System.out.println(action);
        }
    }
    /* Ignore and do not change the code above */
}
```

▶ Voir le code playback

Résultat

Nombreuses boites
Résolution de problèmes +40pts

Seulement des boites STANDARD et REJECTED
Résolution de problèmes +40pts

Seulement des boites STANDARD et SPECIAL
Résolution de problèmes +35pts

Test cas limites
Résolution de problèmes +35pts

Question 2: Somme de facteurs

Language independent | 07:31 / 10:00 | 1x (3 sec) | 200 / 200 pts

Question

La méthode `computeMultiplesSum(n)` doit renvoyer la somme de tous les multiples positifs de 3 ou 5 ou 7 strictement inférieurs à `n`.

Par exemple, pour `n=11`, on obtient 3,5,6,7,9,10 en tant que multiples et la somme de ces multiples vaut 40.

Implémentez `computeMultiplesSum(n)`.

Contraintes:
`0 # n < 1000`

Réponse

JAVA

```java
import java.util.*;
import java.io.*;
import java.math.*;

class Solution {

    public static int computeMultiplesSum(int n) {
        // Write your code here
        // To debug: System.err.println("Debug messages...");
        int sum= 0;
        for (int i = 1; i < n; i++) {
            if(i%3 == 0 || i%5== 0 ||  i%7== 0 )
            sum += i;
        }
        return sum;
    }

    /* Ignore and do not change the code below */
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        PrintStream outStream = System.out;
        System.setOut(System.err);
        int res = computeMultiplesSum(n);
        System.setOut(outStream);
        System.out.println(res);
    }
    /* Ignore and do not change the code above */
}
```

▶ Voir le code playback

Résultat

n=20
Résolution de problèmes +40pts

n=125
Résolution de problèmes +40pts

n=202
Résolution de problèmes +40pts

n=996
Résolution de problèmes +40pts

n=0
Fiabilité +40pts

Question 3: Portails

Language independent | 06:48 / 20:00 | 5x (56 sec) | 25 / 150 pts

Question

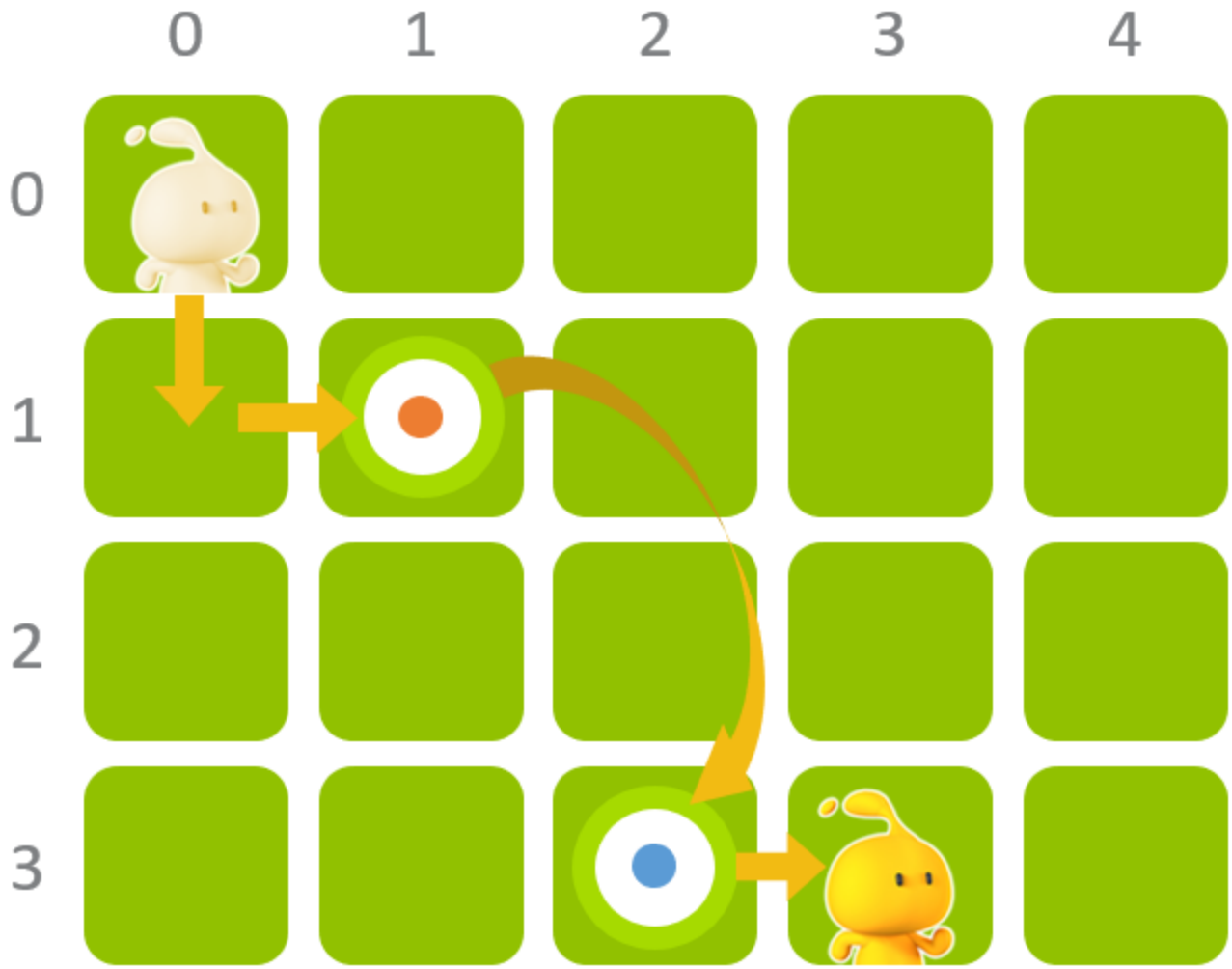
Objectif
Votre avatar de jeu évolue dans un monde étrange comportant deux portails interspatiaux et bidirectionnels. Écrivez un programme retournant les coordonnées de votre avatar compte tenu d'une série de déplacements et de l'emplacement des portails.

Fonctionnement Le terrain est représenté par une grille de `width` cases de large et de `height` cases de haut. La case en haut à gauche est située à `(0, 0)` où le premier entier représente la colonne et le second la ligne. Les positions initiales de votre avatar et des deux portails sont données par des tableaux de deux entiers `position`, `portalA`, et `portalB`. La série de déplacement, `moves`, est une chaîne composée des caractères `U` (haut), `D` (bas), `R` (droite), `L` (gauche).

Si votre avatar marche vers une case comportant un portail, il se téléporte au portail associé (et il reste sur cette case cible tant qu'il n'effectue pas d'autre déplacement). S'il bute contre une extrémité du terrain, il n'avance pas et ne se téléporte pas.

Implémentation
Implémentez la méthode `computeFinalPosition(width, height, position, portalA, portalB, moves)` qui :
prend en entrées les entiers `width`, `height`, les tableaux d'entiers `position`, `portalA`, `portalB`, et la chaîne de caractères `moves` avec : 0 < `width` < 20 0 < `height` < 20 0 <= nombre de caractères de `moves` <= 255 et retourne la position finale de votre avatar sous la forme d'un tableau de deux entiers.

Exemple



Réponse

JAVA

```java
import java.util.*;
import java.io.*;
import java.math.*;

class Solution {

    public static int[] computeFinalPosition(int width, int height, int[] position, int[] portalA, int[] portalB, String moves) {
        // Write your code here
        // To debug: System.err.println("Debug messages...");

        return new int[2];
    }

    /* Ignore and do not change the code below */
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);
        int width = in.nextInt();
        int height = in.nextInt();
        if (in.hasNextLine()) {
            in.nextLine();
        }
        String moves = in.nextLine();
        int[] position = new int[2];
        for (int i = 0; i < 2; i++) {
            position[i] = in.nextInt();
        }
        int[] portalA = new int[2];
        for (int i = 0; i < 2; i++) {
            portalA[i] = in.nextInt();
        }
        int[] portalB = new int[2];
        for (int i = 0; i < 2; i++) {
            portalB[i] = in.nextInt();
        }
        PrintStream outStream = System.out;
        System.setOut(System.err);
        int[] finalPosition = computeFinalPosition(width, height, position, portalA, portalB, moves);
        System.setOut(outStream);
        for (int i = 0; i < 2; i++) {
            System.out.println(finalPosition[i]);
        }
    }
    /* Ignore and do not change the code above */
}
```

▶ Voir le code playback

Résultat

- Déplacements sans portail
  Résolution de problèmes ~~+25pts~~
- Déplacements avec portail
  Résolution de problèmes ~~+25pts~~
- Utilisation d'un portail dans les deux sens
  Résolution de problèmes +25pts
- Déplacements complexes avec murs
  Fiabilité ~~+25pts~~
- Pas de déplacements
  Fiabilité ~~+25pts~~
- Position finale sur un portail
  Fiabilité ~~+25pts~~

Question 4: Javanais

Language independent | 10:25 / 12:00 | 12x (22 sec) | 0 / 100 pts

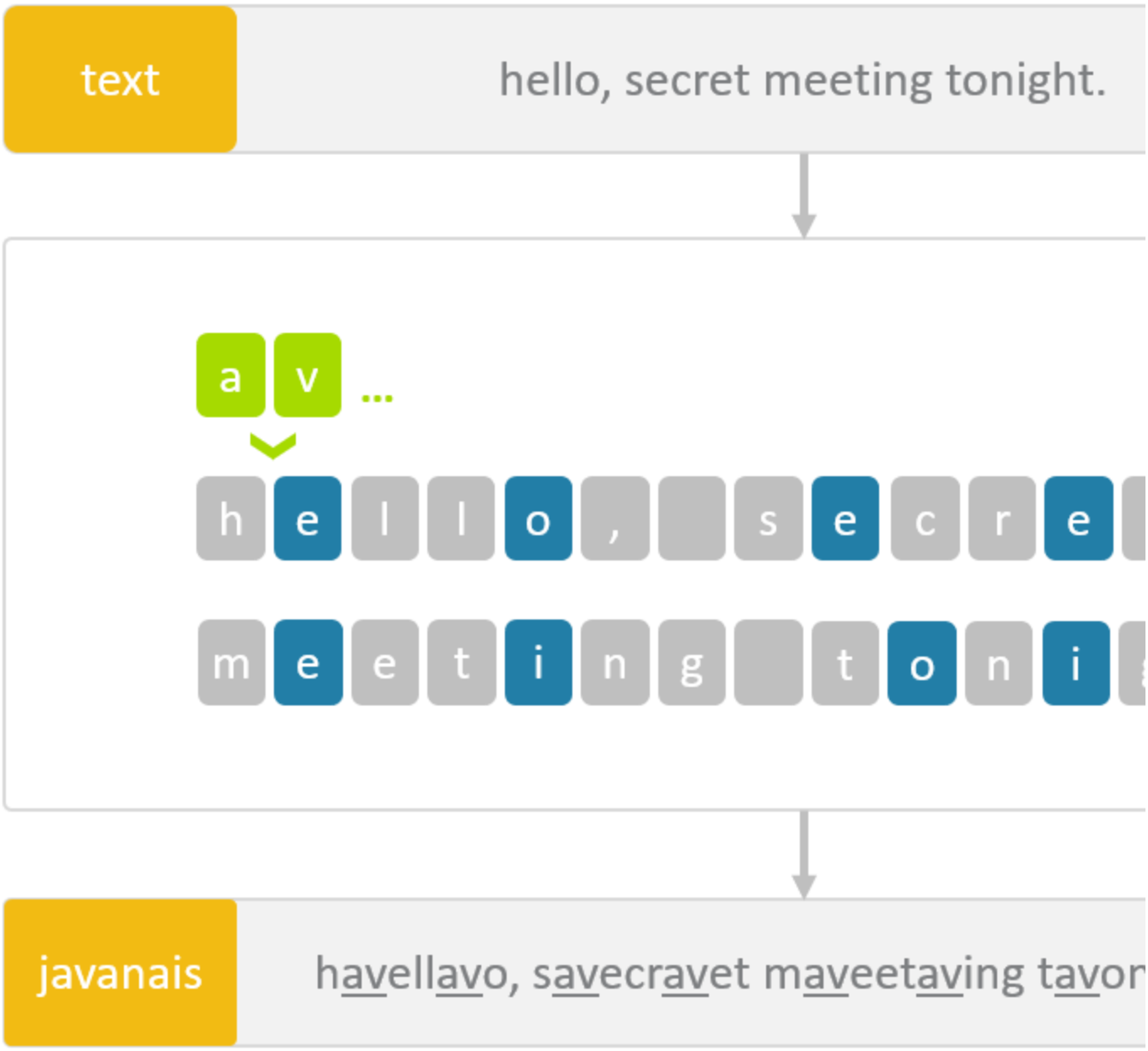
Question

Objectif
Le javanais, aussi appelé langue de feu, est un procédé de codage argotique qui fut utilisé à la fin du 19ème siècle par certains malfaiteurs pour crypter leurs conversations. Écrivez un programme retournant la traduction en javanais d'une phrase.

Fonctionnement Avant chaque voyelle suivante (a, e, i, o, u), insérez la syllable parasite "av" ; Sauf si la voyelle est précédée d'une autre voyelle.

Implémentation
Implémentez la méthode `translate(text)` qui :
prend en entrée `text`, une chaîne de caractères de moins de 255 caractères ; retourne la traduction en javanais, sous la forme d'une chaîne de caractères.
Par simplification, les entrées ne contiennent que des minuscules.

Exemple


text
hello, secret meeting tonight.
a
v
...
h e l l o , s e c r e
m e e t i n g t o n i
javanais
havellavo, savecravet maveetaving tavor

Réponse

JAVA

```java
import java.util.*;
import java.io.*;
import java.math.*;

class Solution {

    public static String translate(String text) {
        String result = ""+text.charAt(0);
        boolean isPreceededByVoyelle = false;
        boolean isPreceededByConsonne = false;
        for(int i = 0; i< text.length() - 1; i++) {

        }
    }

    /* Ignore and do not change the code below */
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);
        String text = in.nextLine();
        PrintStream outStream = System.out;
        System.setOut(System.err);
        String javanais = translate(text);
        System.setOut(outStream);
        System.out.println(javanais);
    }
    /* Ignore and do not change the code above */
}
```

Voir le code playback

Résultat

- Mot simple
  Résolution de problèmes ~~+20pts~~
- Mot avec doubles voyelles
  Résolution de problèmes ~~+20pts~~
- Phrase complexe
  Résolution de problèmes ~~+20pts~~
- Que des voyelles
  Fiabilité ~~+20pts~~
- Que des consonnes
  Fiabilité ~~+20pts~~

Glossaire

Connaissance du langage

La mesure de cette compétence permet de déterminer l'expérience du candidat dans la pratique d'un langage de programmation. **Privilégiez cette compétence si, par exemple, vous recherchez un développeur qui devra être rapidement opérationnel.**

Modélisation

Cette mesure fournit une indication sur la capacité du candidat à appliquer des solutions standard pour résoudre des problèmes récurrents. Un développeur ayant un bon niveau dans cette compétence augmentera la qualité (maintenabilité, évolutivité) de vos applications. Cette compétence ne dépend pas spécifiquement d'une technologie. **Privilégiez cette compétence si, par exemple, vous recherchez un développeur qui sera amené à travailler sur les briques qui structurent vos applications, à anticiper les besoins de demain pour développer des solutions pérennes.**

Résolution de problèmes

Cette compétence correspond aux aptitudes du candidat à comprendre et à structurer son raisonnement pour trouver des solutions à des problèmes complexes. Cette compétence ne dépend pas spécifiquement d'une technologie. **Privilégiez cette compétence si, par exemple, vos applications ont une composante technique importante (R&D, innovation).**

Fiabilité

La fiabilité caractérise la capacité du candidat à réaliser des solutions qui prennent en compte les cas particuliers. Plus cette compétence est élevée, plus vos applications sont robustes (moins de bugs).