Commencé le	mardi 14 décembre 2021, 13:32
État	Terminé
Terminé le	dimanche 2 janvier 2022, 15:00
Temps mis	19 jours 1 heure
Points	3,00/7,00
Note	**8,57** sur 20,00 (**43**%)

Question 1

Correct

Note de 1,00 sur 1,00

Écrire une méthode ecr_b2(int nb) qui n'utilise pas Integer.toBinaryString(nb) et qui retourne comme résultat :

```
String str = Integer.toBinaryString(nb);
```

pour tout nb positif ou nul.

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```
// n est un entier positif ou nul
public static String ecr_b2(int i){
      if (i == 0) {
            return "0";
        } else if (i == 1) {
            return "1";
        } else {
            return  ecr_b2(i / 2) + i % 2;
        }
}
```

	Test	Résultat attendu	Résultat obtenu	
✔	ecr_b2(5)	101	101	✔
✔	ecr_b2(0)	0	0	✔
✔	ecr_b2(1)	1	1	✔
✔	ecr_b2(17)	10001	10001	✔

Tous les tests ont été réussis ! ✔

Solution de l'auteur de la question (Java):

```
public static String ecr_b2(int n){
        if (n<2) return Integer.toString(n%2);
        return ecr_b2(n/2)+ n%2;
}
```

Correct

Note pour cet envoi : 1,00/1,00.

Question 2

Correct

Note de 1,00 sur 1,00

Écrire une méthode eval(String bin) qui n'utilise pas Integer.parseInt et qui retourne l'int nb :

```
    int nb = Integer.parseInt(bin, 2);
```

pour toute écriture binaire bin.

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```
// bin est une écriture binaire
public static int eval(String i){
    int total = 0;
        for (int j = 0 ; j<i.length() ; j++) {
            int compteur = j;
            int valeur = Integer.parseInt(String.valueOf(i.charAt(j)));
            double pui2 = Math.pow(2,i.length()-1-j);
            int calcul = (int) (valeur * pui2);
           // System.out.println("compteur : "+ compteur + " valeur : "+ valeur + " pui

            total += calcul;

        }
        return total;
}
```

	Test	Résultat attendu	Résultat obtenu	
✔	eval("101")	5	5	✔
✔	eval("0")	0	0	✔
✔	eval("1")	1	1	✔
✔	eval("1011")	11	11	✔

Tous les tests ont été réussis ! ✔

Solution de l'auteur de la question (Java):

```
public static int eval(String bin){
    if (bin.equals("0")) return 0;
    if (bin.equals("1")) return 1;
    char dernierChiffre = bin.charAt(bin.length()-1);
    String eSansDernierChiffre = bin.substring(0, bin.length() - 1);
    if (dernierChiffre =='0') return 2*eval(eSansDernierChiffre);
    //le dernier chiffre est 1
    return 2*eval(eSansDernierChiffre)+1;
}
```

Correct

Note pour cet envoi : 1,00/1,00.

Question 3

Correct

Note de 1,00 sur 1,00

Pour que la **méthode** qui est complétée

```
public static String add_b2(String e1, e2)
```

retourne une String qui l'écriture en base 2, de la somme eval(e1) + eval(e2), **écrire la méthode récursive suivante** de signature

```
private static String add_b2(String e1, e2, int carry)
```

qui retourne une String qui l'écriture en base 2, de la somme eval(e1) + eval(e2) + carry.

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```java
public static String add_b2(String e1, String e2){
        String string_e1 = e1;
        String string_e2 = e2;
        for (int e1_condition = e1.length(); e1_condition < e2.length(); e1_condition++
            string_e1 = "0" + string_e1;
        }

        for (int e2_condition = e2.length(); e2_condition < e1.length(); e2_condition++
            string_e2 = "0" + string_e2;
        }

        return add_b2(string_e1, string_e2,0);
    }

    private static String add_b2(String e1, String e2, int carry) {
        if (e1.isEmpty() || e2.isEmpty()) {
            if (carry == 1) {
                return "1";
            } else {
                return "";
            }

        } else {
            String e1_char = String.valueOf(e1.charAt(e1.length() - 1));
```

	Test	Résultat attendu	Résultat obtenu	
✔	add_b2("101","100")	1001	1001	✔
✔	add_b2("101","10")	111	111	✔
✔	add_b2("111","1")	1000	1000	✔
✔	add_b2("101","0")	101	101	✔
✔	add_b2("0","0")	0	0	✔
✔	add_b2("1","1")	10	10	✔
✔	add_b2("01","01")	10	10	✔

Tous les tests ont été réussis ! ✔

Solution de l'auteur de la question (Java):

```java
    /**
     *
     * @param e1
     * @param e2
     * @return
     */
    public static String add_b2(String e1, String e2){
        return add_b2(e1,e2,0);
    }
```

```java
    private static String add_b2(String e1, String e2, int carry){
        if (e1.equals("") && e2.equals("") && (carry == 0)) return "";
        if (e1.equals("") && e2.equals("")) return "1";
        if ((e1.equals("") || e2.equals("")) && (carry==0)) return e1+e2;
        if (e1.equals("") || e2.equals("")) return add_b2(e1+e2,"1",0);
        // !e1.equals("") & !e2.equals("")
        char c1 = e1.charAt(e1.length()-1);
        char c2 = e2.charAt(e2.length()-1);
        int s = (c1-'0')+(c2-'0')+carry;
        return add_b2(e1.substring(0,e1.length()-1), e2.substring(0,e2.length()-1),s/2)
    }
```

Correct

Note pour cet envoi : 1,00/1,00.

Question 4

Incorrect

Note de 0,00 sur 1,00

Pour que la **méthode qui est complétée**

```
public static String c_add_b2(String e1, e2)
```

retourne une String qui est la ligne des retenues de la somme eval(e1) + eval(e2), avec '-' quand la retenue est 0.

La méthode suivante qui est contenue dans la classe de test

```
affiche(e1,e2)
```

affiche :
e1
+e2
ligne des retenues
add_b2(e1,e2)

La méthode add_b2 est également écrite dans la classe de test.

Écrire la méthode récursive suivante de signature

```
private static String c_add_b2(String e1, e2, int carry)
```

Par exemple:

Test	Résultat
`affiche("101","100")`	101 +100 1--- 1001
`affiche("101","10")`	101 + 10 ---- 111

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```
1  /**
2   *
3   * @param e1
4   * @param e2
5   * @return
6   */
7  public static String c_add_b2(String e1, String e2){
8      return c_add_b2(e1,e2,0)+"-";
9  }
10
11 private static String c_add_b2(String e1, String e2, int carry){
12     return "";
13 }
```

	Test	Résultat attendu	Résultat obtenu	

	Test	Résultat attendu	Résultat obtenu	
✖	affiche("101","100")	101 +100 1--- 1001	101 +100 - 1001	✖
✖	affiche("101","10")	101 + 10 ---- 111	101 + 10 - 111	✖
✖	affiche("111","1")	111 + 1 111- 1000	111 + 1 - 1000	✖
✖	affiche("101","0")	101 + 0 ---- 101	101 + 0 - 101	✖
✖	affiche("0","0")	0 +0 -- 0	0 +0 - 0	✖
✖	affiche("1","1")	1 +1 1- 10	1 +1 - 10	✖
✖	affiche("01","01")	01 +01 -1- 10	01 +01 - 10	✖
✖	affiche("101","1")	101 + 1 --1- 110	101 + 1 - 110	✖

Votre code doit réussir tous les tests pour gagner des points. Recommencer.

Montrer les différences

Solution de l'auteur de la question (Java):

```java
    /**
     *
     * @param e1
     * @param e2
     * @return
     */
    public static String c_add_b2(String e1, String e2){
        return c_add_b2(e1,e2,0)+"-";
    }

    private static String c_add_b2(String e1, String e2, int carry){
        if (e1.equals("") && e2.equals("")) return "";
        if (e1.equals("") || e2.equals("")) return c_add_b2(e1+e2,""+carry,0);
        char c1 = e1.charAt(e1.length()-1);
        char c2 = e2.charAt(e2.length()-1);
        int s = (c1-'0')+(c2-'0')+carry;
        String c = "-";
        if (s > 1) c = "1";
        return c_add_b2(e1.substring(0,e1.length()-1), e2.substring(0,e2.length()-1),s/
    }
```

Incorrect

Note pour cet envoi : 0,00/1,00.

Question 5

Incorrect

Note de 0,00 sur 1,00

Ajoutez à la classe suivante :

```
public class PostFixe {
  private String[] uniteLex;
  private int indice = 0;

  private String getUniteLex(){
     return uniteLex[indice++];
  }
  private boolean isNumber(String s){
     try{ Integer.parseInt(s); }
     catch(NumberFormatException e)
     { return false; }
     return true;
  }

  private int eval(String op, int u, int v){
    switch(op){
      case "+": return u + v;
      case "*": return u * v;
      case "/": return u / v;
      case "-": return u - v;
      default: throw new RuntimeException("op inconnu");
      }
  }

  public void setPostFixe(String expression) {
      this.uniteLex = expression.split(" ");
  }
}
```

la méthode

```
public int eval()
```

qui évalue l'expression polonaise (supposée syntaxiquement correcte) à laquelle elle est appliquée :

Par exemple:

Test	Résultat
`this.setPostFixe("+ + 1 + 2 3 + 4 5");` `System.out.print(this.eval());`	15
`this.setPostFixe("+ * 2 3 10");` `System.out.print(this.eval());`	16
`this.setPostFixe("+ - 8 7 20");` `System.out.print(this.eval());`	21
`this.setPostFixe("/ 16 + 3 5");` `System.out.print(this.eval());`	2

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```
1  // compléter la methode eval
2  public int eval() {
3       return 0;
4  }
```

	Test	Résultat attendu	Résultat obtenu	
✖	`this.setPostFixe("+ + 1 + 2 3 + 4 5");` `System.out.print(this.eval());`	15	0	✖
✖	`this.setPostFixe("+ * 2 3 10");` `System.out.print(this.eval());`	16	0	✖
✖	`this.setPostFixe("+ - 8 7 20");` `System.out.print(this.eval());`	21	0	✖
✖	`this.setPostFixe("/ 16 + 3 5");` `System.out.print(this.eval());`	2	0	✖

Votre code doit réussir tous les tests pour gagner des points. Recommencer.

Montrer les différences

Solution de l'auteur de la question (Java):

```java
    public int eval() {
        String uniteLex = getUniteLex();
        if (isNumber(uniteLex)) return Integer.parseInt(uniteLex);
        return eval(uniteLex, eval(), eval());
    }

```

Incorrect

Note pour cet envoi : 0,00/1,00.

Question 6

Incorrect

Note de 0,00 sur 1,00

Ajoutez à la classe suivante :

```
public class PostFixe {
  private String[] uniteLex;
  private int indice = 0;

  private String getUniteLex(){
     return uniteLex[indice++];
  }
  private boolean isNumber(String s){
     try{ Integer.parseInt(s); }
     catch(NumberFormatException e)
     { return false; }
     return true;
  }

  private String evalToECP(String op, String u, String v){
     return "("+u+op+v+")" ;
  }

  public void setPostFixe(String expression) {
      this.uniteLex = expression.split(" ");
  }
}
```

la méthode

```
public String evalToECP()
```

qui traduit en une expression complètement parenthésée, l'expression polonaise (supposée syntaxiquement correcte) à laquelle elle est appliquée :

Par exemple:

Test	Résultat
this.setPostFixe("+ + 1 + 2 3 + 4 5"); System.out.print(this.evalToECP());	((1+(2+3))+(4+5))
this.setPostFixe("+ * 2 3 - 10 9"); System.out.print(this.evalToECP());	((2*3)+(10-9))
this.setPostFixe("+ - 8 7 20"); System.out.print(this.evalToECP());	((8-7)+20)
this.setPostFixe("/ 16 + 3 5"); System.out.print(this.evalToECP());	(16/(3+5))

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```
1  // compléter la methode eval
2  public String evalToECP() {
3       return "";
4  }
```

	Test	Résultat attendu	
✖	`this.setPostFixe("+ + 1 + 2 3 + 4 5");` `System.out.print(this.evalToECP());`	`((1+(2+3))+(4+5))`	✖
✖	`this.setPostFixe("+ * 2 3 - 10 9");` `System.out.print(this.evalToECP());`	`((2*3)+(10-9))`	✖
✖	`this.setPostFixe("+ - 8 7 20");` `System.out.print(this.evalToECP());`	`((8-7)+20)`	✖
✖	`this.setPostFixe("/ 16 + 3 5");` `System.out.print(this.evalToECP());`	`(16/(3+5))`	✖

Votre code doit réussir tous les tests pour gagner des points. Recommencer.

Solution de l'auteur de la question (Java):

```java
    public String evalToECP() {
        String uniteLex = getUniteLex();
        if (isNumber(uniteLex)) return uniteLex;
        return evalToECP(uniteLex, evalToECP(), evalToECP());
    }

```

Incorrect

Note pour cet envoi : 0,00/1,00.

Question 7

Incorrect

Note de 0,00 sur 1,00

Ajoutez à la classe suivante :

```
public class PostFixe {
  private String[] uniteLex;
  private int indice = 0;

  private String getUniteLex(){
     return uniteLex[indice++];
  }
  private boolean isBool(String s){
     return s.equals("true") || s.equals("false");
  }

  private boolean eval(String op, boolean u, boolean v){
    switch(op){
      case "&": return u && v;
      case "|": return u || v;
      case "!": return !u;
      default: throw new RuntimeException("op inconnu");
      }
  }

  public void setPostFixe(String expression) {
      this.uniteLex = expression.split(" ");
  }
}
```

la méthode

```
public boolean eval()
```

qui évalue l'expression polonaise booléenne (supposée syntaxiquement correcte) à laquelle elle est appliquée :
Par exemple:

Test	Résultat
`this.setPostFixe("! false");` `System.out.print(eval());`	`true`
`this.setPostFixe("| & true true true");` `System.out.print(this.eval());`	`true`
`this.setPostFixe("| true false");` `System.out.print(this.eval());`	`true`
`this.setPostFixe("& true false");` `System.out.print(this.eval());`	`false`

Réponse : (régime de pénalités : 0 %)

Réinitialiser la réponse

```
1  // compléter la methode eval
2  public boolean eval() {
3       return false;
4  }
```

	Test	Résultat attendu	Résultat obtenu	
✖	`this.setPostFixe("! false");` `System.out.print(eval());`	true	false	✖
✖	`this.setPostFixe("| & true true true");` `System.out.print(this.eval());`	true	false	✖
✖	`this.setPostFixe("| true false");` `System.out.print(this.eval());`	true	false	✖
✔	`this.setPostFixe("& true false");` `System.out.print(this.eval());`	false	false	✔
✔	`this.setPostFixe("& & true & true false & true true");` `System.out.print(this.eval());`	false	false	✔
✔	`this.setPostFixe("| ! true false");` `System.out.print(this.eval());`	false	false	✔
✔	`this.setPostFixe("! | true false");` `System.out.print(this.eval());`	false	false	✔

Votre code doit réussir tous les tests pour gagner des points. Recommencer.

Montrer les différences

Solution de l'auteur de la question (Java):

```
1   public boolean eval() {
2       String uniteLex = getUniteLex();
3       if (isBool(uniteLex)) return Boolean.parseBoolean(uniteLex);
4       boolean e1 = eval();
5       boolean e2 = true;
6       if (uniteLex.equals("!")) e2 = eval();
7       return eval(uniteLex, e1, e2);
8   }
9
```

◄ QCM_entrainement_def_inductives_fcts_prog

Aller à...

Modalités de fonctionnement des contrôles (sous réserve de modifications dont vous seriez informés) ►

Incorrect

Note pour cet envoi : 0,00/1,00.