

R2.03 TD1

Table des matières

| I. Exercice 1 | 2 |
|----------------------------|---|
| 1. Question 1 | 2 |
| 1. Question 1 | 2 |
| 3. Question 3 | 2 |
| 4. Ouestion 4 | 2 |
| 4. Question 4 | 2 |
| 6. Ouestion 6 | 2 |
| 6. Question 67. Question 7 | 2 |
| 8. Question 8 | : |
| II. Exercice 2 | |
| II. Exercice 2 | |
| 2. Question 2 | |
| 3. Question 3 | 4 |
| 4. Question 4 | |
| 5. Question 5 | |
| III. Exercice 3 | |

I. Exercice 1

1. Question 1

res1 = 15

res2 = 14348907

2. Question 2

res1 = 40

res2 = 1099511627776

3. Question 3

res1 = 7

res2 = 128

4. Question 4

Fait.

5. Question 5

Cela ne s'arrête plus aux breakpoints et donc exécute.

6. Question 6

Fait. Une fois avoir fait step into on se retrouve dans l'autre classe et en faisant step return on retourne à l'endroit d'origine.

7. Question 7

| i | terme | ret |
|---|-------|-----|
| 1 | 1 | 1 |
| 2 | 2 | 3 |
| 3 | 4 | 7 |
| 4 | 8 | 15 |

La somme des termes multiplié par l'exposant, sachant que le nouveau terme est l'ancien terme multiplié par l'exposant et que le terme de base est 1.

Oui les deux expressions sont true donc invariant. La formule mathématique est U1 = 1 avec Un+1 = 2Un + 1. Dans l'exemple utilisé on cherche U4.

8. Question 8

| i | р | ret |
|---|----|-----|
| 3 | 31 | 27 |
| 4 | 31 | 81 |

A voir comme ça, je constate qu'on a 3 * 27 = 81. Et 3 est la base j'en déduis donc qu'après ce sera 81 * 3 et ainsi de suite.

L'expression pour la valeur finale de ret avec base et p est ret==Math.pow(this.base,p) c'est-à-dire que ret = base ** (Urang) sachant que Un+1 = (exposant * Un) + 1

Q1 : rang = 3, ba = 3, $\exp = 2$ donc U3 = 15 = res1 et 3 ** 15 = 14348907 = res2

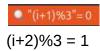
Q2 : rang = 3, ba = 2, exp = 3 donc U3 = 40 = res1 et 2 ** $40 = 1,099511628 \times 10^{12} = res2$

Q1 : rang = 2, ba = 2, exp = 2 donc U3 = 7 = res1 et 2 ** 7 = 128 = res2

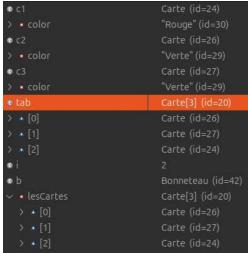
II. Exercice 2

1. Question 1





2. Question 2



Tab et lesCartes ont le même id car juste avant on appelle le constructeur qui copie l'adresse de tab dans l'attribut les Cartes.

 • c1
 Carte (id=24)

 > • color
 "Rouge" (id=30)

 • c2
 Carte (id=26)

 > • color
 "Verte" (id=29)

 • c3
 Carte (id=27)

 > • color
 "Verte" (id=29)

 • tab
 Carte[3] (id=20)

 > • [0]
 Carte (id=26)

 > • [1]
 Carte (id=27)

 • i
 2

 • b
 Bonneteau (id=42)

 ∨ • lesCartes
 Carte[3] (id=20)

 > • [0]
 Carte (id=26)

 > • [1]
 Carte (id=24)

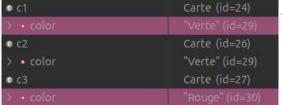
 > • [2]
 Carte (id=27)

Prévision de b.swap(...); Après avoir exécuter on retrouve la prévision

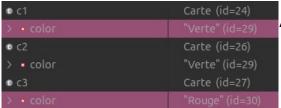
3. Question 3



Avant prévision



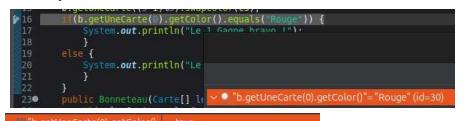
Après prévision



Après exécution de la ligne de commande

Le 1 perd désolé !

4. Question 4



L'utilisateur va gagner

b.getUneCarte(0).getColor().equals("Rouge") renvoie true s'il va gagner



5. Question 5



Je change la couleur de la carte à l'indice 0 directement dans les variables

III. Exercice 3

$t = \{4,1,1,3,4\};$

l.26 : Index out of bounds car on utilise comme indice la taille, hors la taille
n'est pas le dernier indice. Il faut mettre < à la place de <= dans la condition
de continuation de la boucle.</pre>

for (int j=i;j<=this.tab.length;j++) → for (int j=i;j<this.tab.length;j++){</pre>

l.26 : Le numéro que l'on compare avec les autres se compare avec lui même alors qu'on le met déjà à 1, signe de déjà le compter. Il faut mettre j=i+1 au lieu de j=i.

for (int j=i;j<this.tab.length;j++){ \rightarrow for (int j=i+1;j<this.tab.length;j++)

l.31 : Si le nombre d'occurrences du nombre actuelle est plus grand ou égal à au nombre d'occurrences maximum alors on change, hors dans le cas actuelle 4 et 1 sont tout deux au même nombre d'occurrences mais on cherche le premier alors. Mais avec le >= on prend le dernier car si égal on change. Il faut donc remplacer >= par >.

if $(nb0cc >= nbMax) \{ \rightarrow if (nb0cc > nbMax) \}$

1.33 : Le programme met dans ret l'indice où se trouve la première valeur avec le plus grand nombre d'occurrences, hors nous voulons directement cette valeur. <u>Il fau</u>t d<u>onc changer</u> ret=i par ret=tab[i].

ret=i; → ret=tab[i];

Le nombre maximum d'occurence est :2 4

Maintenant que le programme est débugger on retrouve le résultat attendu pour le tableau $\{4,1,1,3,4\}$.

Pour un tableau vide tout va bien aussi.

Mais pour le tableau null, non.

1.22 : Dans la condition, on cherche à savoir la taille du tableau avant de savoir si il est null. Il faut inverser ces deux conditions.

if ((this.tab.length>0)&&(this.tab!=null)) → if ((this.tab!=null)&&(this.tab.length>0))