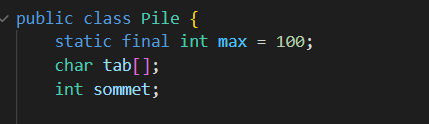
**Compte rendu de JAVA**

**Exercice 1 :**

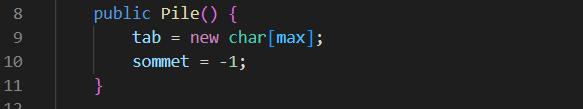


Dans cette partie du code j’ai déclaré une classe appelée **Pile**. J’ai aussi déclaré 3 variable :

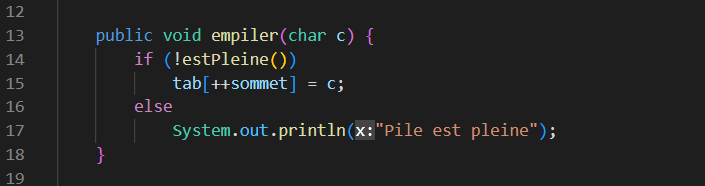
**Max** : est une constante représentant la taille maximale de la pile (100 dans ce cas).

**Tab** : est un tableau de caractères utilisé pour stocker les éléments de la pile.

**Sommet** : est un indice qui représente le sommet de la pile.

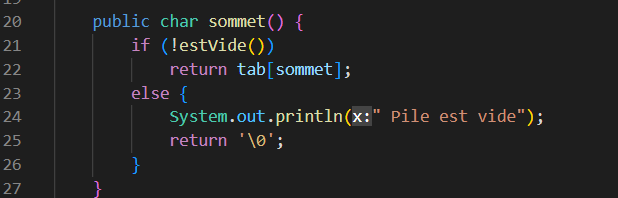


Dans cette partie j’ai utilisé un constructeur initialise le tableau **tab** avec une nouvelle instance de tableau de caractères de taille **max** et initialise **sommet** à -1, indiquant que la pile est vide.



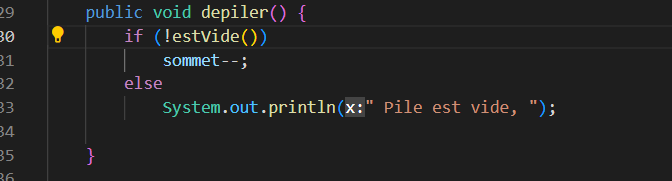
J’ai utilise la méthode **empiler(char c)** pour ajouter un caractere **c** au sommet de la pile si la pile n'est pas pleine.

Sinon, elle affiche un message indiquant que la pile est pleine.

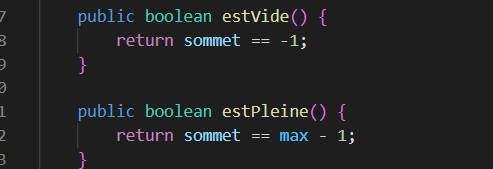


Cette méthode renvoie le **sommet** de la pile s'il y a des éléments dans la pile.

Sinon, elle affiche un message indiquant que la pile est vide et renvoie le caractère nul '\0'.



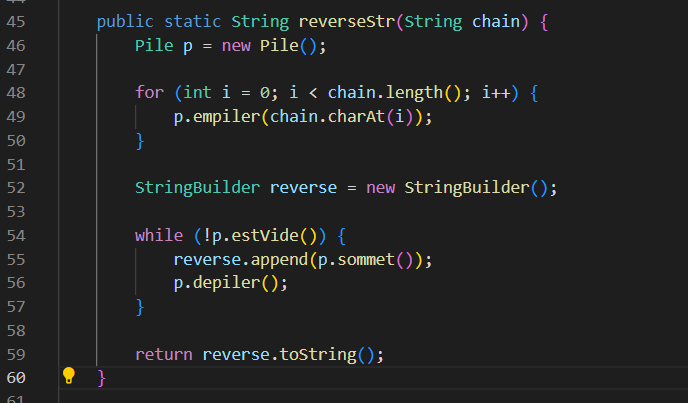
Cette méthode est utilisee pour retirer l'élément du sommet de la pile s'il y a des éléments dans la pile. Sinon, elle affiche un message indiquant que la pile est vide.



Dans cette partie j’ai utilisé 2 méthodes :

La première méthode renvoie **true** si la pile est vide (si sommet est égal à -1), sinon elle renvoie **false.**

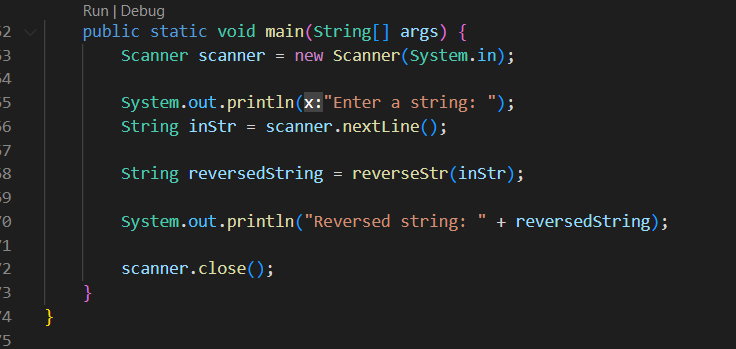
La deuxième renvoie **true** si la pile est pleine (si sommet est égal à max - 1), sinon elle renvoie **false**.



J’ai utilisé la méthode **reverseStr(String chain)** qui prend une chaine de caractères en entrée et renvoie l’inverse de cette chaine a l’aide d’une pile.

Au début. J’ai créé une instance de classe Pile, après j’ai utilisé une boucle pour parcourir la chaine et empile les caractères sut la pile p. J’ai créé un objet **StringBuilder** pour construire la chaîne inversée de manière efficace. Puis Tant que la pile n'est pas vide, le caractère du sommet de la pile est ajouté à reverse, puis la pile est dépilée. Cela se fait jusqu'à ce que la pile soit vide.

Enfin, la méthode retourne la version inversée de la chaîne en appelant la méthode toString() sur reverse.



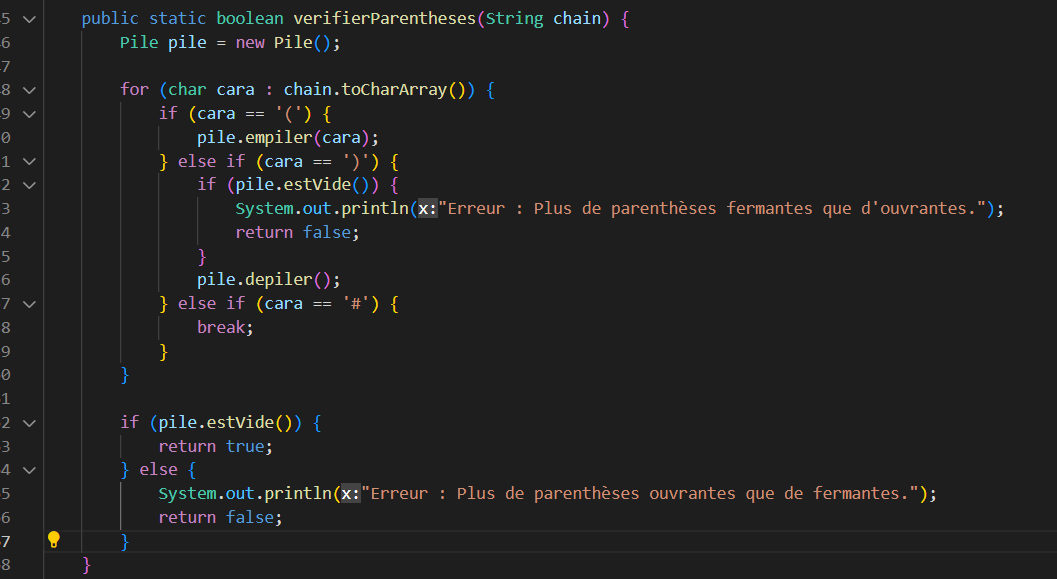
La fonction main() lire une chaine de caractères est utiliser la méthode reverseStr() avec la chaine entrée par l’utilisateur ,et stocker l’inverse dans le variable reversedString, après elle affiche la chaine inverse dans l’écran.

Résultat du code :

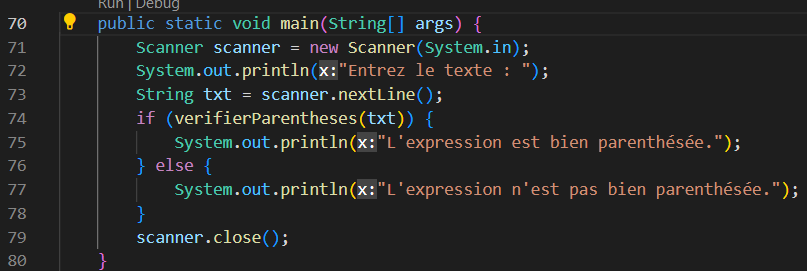


**Exercice 2 :**

Premièrement, j’ai utilisé les méthodes empiler(), depiler(), estVide(), estPleine() et sommet() .



Après, j’ai utilisé dans la méthode **verifierParantheses()** une pile pour suivre les parenthèses dans la chaîne et vérifie si elles sont correctement équilibrées. Si elles le sont, elle renvoie **true**, sinon, elle renvoie **false** avec un message d'erreur approprié.



Enfin, J’ai demandé à l’utilisateur d’entrée un texte, puis je fais une lecture et appeler la méthode verifierParentheses() avec le texte comme argument. Si la méthode retourne true, un message indiquant que l'expression est bien parenthésée est affiché. Sinon, un message indiquant que l'expression n'est pas bien parenthésée est affiché.

Résultat du code :

