

Soutenance

Projet Génie Logiciel - Équipe 07

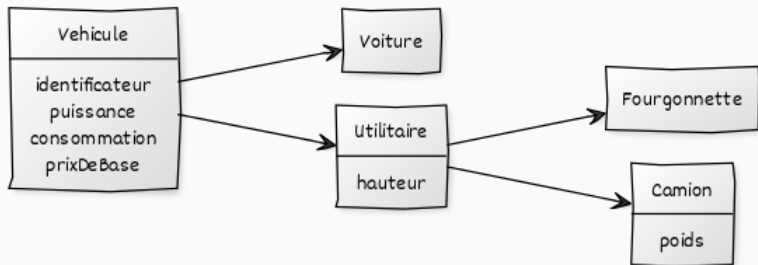
27 janvier 2022

Grenoble INP - Ensimag, UGA

INTRODUCTION

DÉMONSTRATION GARAGISTE

Schéma



- Méthodes héritées vs Méthodes redéfinies

```
float calculerCout() {  
    return prixDeBase + 2*puissance;  
}
```

```
void description() {  
    println(" identificateur = ", identificateur);  
    println(" puissance = ", puissance, " chevaux");  
    println(" consommation = ", consommation, "L/100km");  
    println(" cout = ", calculerCout(), " euros");  
}
```

COMPILATEUR DE BASE

VALIDATION

- Tests boîte-noire \Rightarrow Aspect Global
- Tests unitaires \Rightarrow Validation code Java
- Tests de conformité \Rightarrow Validation des détails
- Tests système \Rightarrow Utilisation réelle
























Scripts Shell

Fonctions réutilisables

```
[RECAP]
[PARSER INVALID TESTS] Results : 11/11
[PARSER VALID TESTS] Results : 13/13
[PARSER ORACLE INVALID TESTS] Results : 292/292
[PARSER ORACLE VALID TESTS] Results : 114/114
[PARSER TOTAL] Results : 430 / 430
```

- Rapport Jacoco
- Tests Unitaires

Deca Compiler

Element	Missed Instructions	Cov.	Missed Branches	Cov.
 fr.ensimag.deca		82%		76%
 fr.ensimag.deca.codegen		93%		83%
 fr.ensimag.deca.context		88%		86%
 fr.ensimag.deca.syntax		75%		55%
 fr.ensimag.deca.tools		94%		100%
 fr.ensimag.deca.tree		94%		89%
 fr.ensimag.ima.pseudocode		78%		75%
 fr.ensimag.ima.pseudocode.instructions		69%		n/a
Total	3,822 of 24,816	84%	475 of 1,579	69%

- *Coding-style*
- Javadoc
- Programmation défensive
- Architecture

EXTENSION

Produit de matrices

Exercice 1 ★ - Interprétation du produit matriciel [Signaler une erreur] [Ajouter à ma feuille d'exos]

Énoncé ▼

Une entreprise désire fabriquer de nouveaux jouets pour Noël : une poupée B et une poupée K. Elle désire commander les matières premières nécessaires pour la fabrication de ces jouets. On dispose des informations suivantes :

- La fabrication d'une poupée B nécessite 0,094kg de coton biologique, 0,2kg de plastique végétal et 0,4kg de pièces métalliques.
- La fabrication d'une poupée K nécessite 0,08kg de coton biologique, 0,3kg de plastique végétal et 0,1kg de pièces métalliques.

Par ailleurs, l'entreprise a réalisé les prévisions de ventes suivantes :

- elle pense vendre 1000 poupées B et 800 poupées K en novembre;
 - elle pense vendre 2500 poupées B et 1200 poupées K en décembre.
1. Disposer les informations obtenues sous la forme de deux tableaux.
 2. En effectuant un produit matriciel, déterminer la quantité de coton biologique à commander pour le mois de décembre, la quantité de plastique végétal pour le mois de novembre.

Initialisation des matrices

Produit matriciel

Accès aux composantes

Cast du résultat

Processus de création (modifications des grammairales)

Recherches bibliographiques

[Ajout non-terminal]
array_creator_expr
→ 'new' ident dim_expr

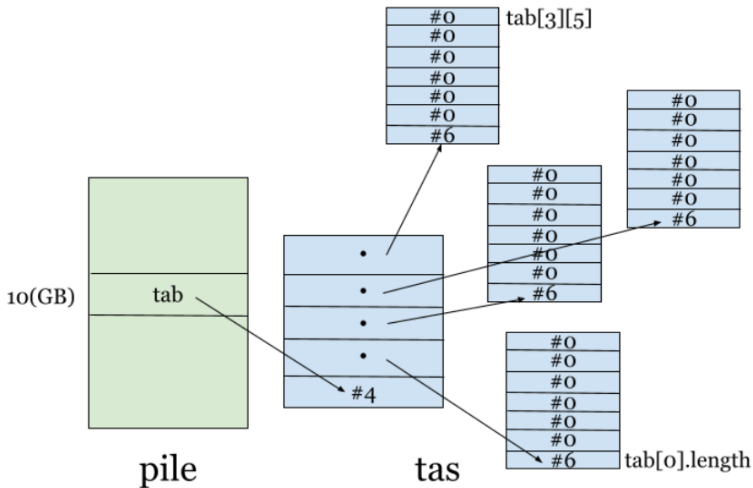
[Ajout non-terminal]
dim_expr
→ '[' **expr** ']' ('[' **expr** ']')*

EXPR → NewArray [IDENTIFIER LIST_EXPR]

expr ↓env_types ↓env_exp ↓class ↑type
→ ...
→ NewArray [
 type ↓env_types ↑type
 {dimension := 0}
 [(**expr_index** ↓env_types ↓env_exp ↓class {dimension := dimension + 1}) *]
]
condition type ∈ {int, float} ET dimension ∈ {1, 2}.

affection type :=
$$\begin{cases} \underline{\text{int}}[] & \text{si type} = \underline{\text{int}} \text{ et dimension} = 1 \\ \underline{\text{int}}[][] & \text{si type} = \underline{\text{int}} \text{ et dimension} = 2 \\ \underline{\text{float}}[] & \text{si type} = \underline{\text{float}} \text{ et dimension} = 1 \\ \underline{\text{float}}[][] & \text{si type} = \underline{\text{float}} \text{ et dimension} = 2 \end{cases}$$

Processus de création (génération de code)



```
int[][] tab = new int[4][6]
```


- Création de matrices/vecteurs uniformes
 - Affichage
 - Opérations binaires (somme, différence, produit)
 - Transposée
1. **boolean isSquarred(float[][] mat)**
Retourne true si la matrice mat est carrée, false sinon.
 2. **float[][] multMat(float[][] mat1, float[][] mat2)**
Retourne une matrice issue de la multiplication des matrices

Rappel du must have :

- Bibliothèque de calcul matriciel
- Tableaux d'int et de float
- Initialisation, accès aux éléments similaire à Java

Limitations :

- Compilateur
- Extension

- Compilateur (lexer, parser, analyse contextuel et génération de code) fonctionnel
- Tableaux et bibliothèque de calcul matriciel

Quoi d'autre maintenant ?

- Davantage tester la génération de code
- Rajouter des tableaux n-dimensionnels de type quelconque
- Fournir de nouvelles bibliothèques Deca

BILAN DE PROJET

- Méthode agile SCRUM
- Outil pour les sprints : *Trello*
- Outil pour le versionnage : *GitLab*

Organisation des rôles

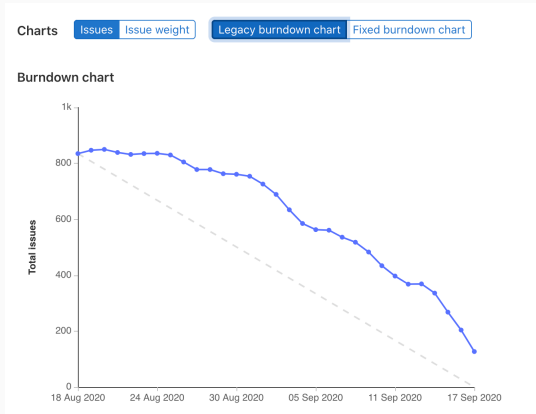
- Scrum Master (1p)
- Développeurs (2p)
- Testeurs (2p)
- Responsable Git (1p)

Division du temps de travail

	Sprint 1	Sprint 2	Sprint 3	Total
Total Analyse	30	28	15	73
Total Développement	40.5	47	52	139.5
Total Validation	59	60.5	48	167.5
Total Documentation	15	11	15	41
Total Extension	4	31	30	65
Total Analyse Énergétique	4	5	8	17

Figure 1: Temps passé (en heures) sur chaque tâche du projet.

- Outils *GitLab*
- Répartition des rôles
- "La réflexion avant l'action"



BILAN D'ÉQUIPE
