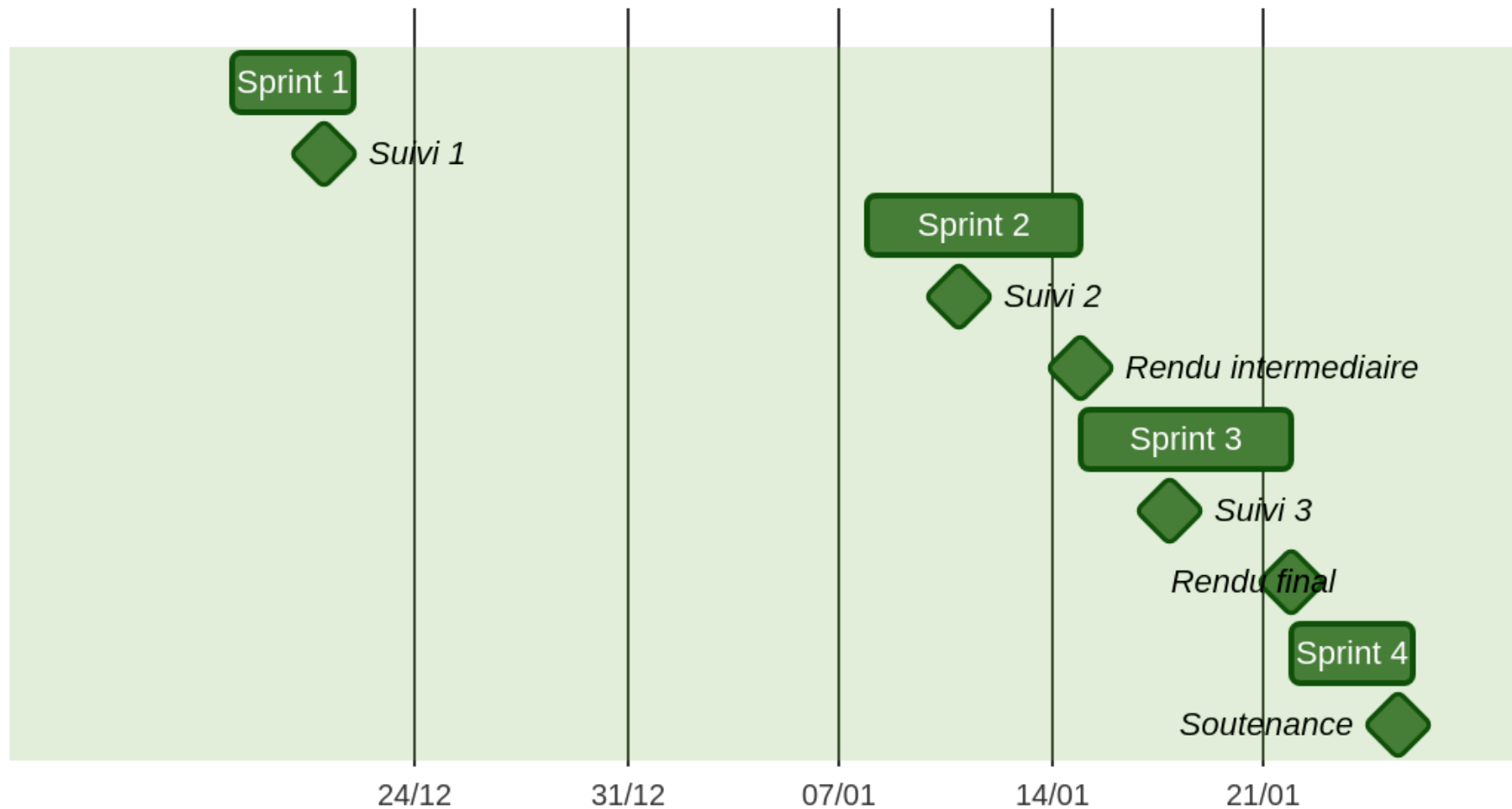


## Planning effectif (Réalisation)



Avec le peu de progression effectif que nous avons réalisé durant les vacances, nous avons décidé de ne pas inclure ces modifications dans le Sprint 2 mais de les considérer comme des contributions volontaires. Ainsi, le sprint 2 se déroule du 08 au 15 janvier et est plus uniforme avec les autres sprints (durée plus ou moins 1 semaine).

# Introduction

Nous avons utilisé les outils internes de GitLab pour contrôler notre progression des User Story. Ainsi, notre planning effectif est une liste de tâches sur un board GitLab mise à jour quotidiennement. Nous encourageons le lecteur à voir la liste de nos Merges Requests sur le dépôt Git pour avoir une meilleure estimation des assignés d'une tâche. En effet, comme nous pouvons avoir un User Story regroupant plusieurs sous-tâches (que ce soit une fonctionnalité sous-jacent, une étape A, B, C, etc), nous utilisons plus les auteurs des merges requests correspondant à ces sous-tâches pour déterminer ce qu'un collègue fait effectivement dans le User Story.

# Sprint 1

Issues 10 Merge requests 6 Participants 6 Labels 2

Unstarted Issues (open and unassigned)	0	Ongoing Issues (open and assigned)	0	Completed Issues (closed)	10
				Fix quotes	#12 Fix
				Ajouter des tests pour Afficher "Hello, World!"	#11 Fix
				Ajout des fonctionnalité -p -v -d -P (pour les println)	#10 User story
				Verify program for hello world	#9 Fix
				Utiliser decac sans option et avec -b	#8 User story
				Participer à l'amphi Git avancé	#6
				Concevoir le planning prévisionnel	#5
				Valider l'extension OPTIM	#4
				Compiler le programme vide	#3 User story
				Afficher "Hello, World!"	#2 User story

## Sprint 2

**Issues 35** Merge requests 34 Participants 5 Labels 5

Unstarted Issues (open and unassigned)

0

Ongoing Issues (open and assigned)

0

Completed Issues (closed)

35

Etablir la documentation de l'extension OPTIM

#21 **Priorité 1** **User story**

Afficher des types int, float

#13 **Priorité 1** **User story**    

op boolean

#50 **Doing**

TEST\_FUN\_IF\_ELSE\_WHILE\_COMPARISON

#49 **Technical Story** 

Ajouts des derniers tests de la soirée

#48 **Technical Story** 

Fix les comparaisons qui sont cassées

#47 **Fix** 

Add new Tests pour l'étape A et B

#45 **Technical Story** 

convFloat pour les opérations arithmétiques

#44 **User story** 

Refactoriser codeGenPrint pour avoir un seul appel WXX

#43 **Fix** 

Vérifier l'affectation avec la nouvelle algo des opérations arithmétiques

#42 **Fix** 

Adapter tout le codegen des expressions binaires à l'algo du prof

#41 

Add tests for arithmetic operations and Etape A

#40 Technical Story 🐛

Deleguer toutes les instructions d'impression a  
AbstractPrint

#39 Fix 🐛

Ajouter sécoration convFloat pour assignment et decl

#38

codeGenPrint pour assignment

#36

Gérer la priorité des opérations arithmétiques selon la  
vidéo des profs

#35 Fix 🐛

Point remonté après les gros merge

#34 Fix 🐛 🐛 🐛

Gestion de READINT et READFLOAT

#33 🐛 🐛

Include

#30 User story 🐛 🐛

Ajouter des tests

#29 🐛

Correction des registres : le boss est R2 normalement

#28 Fix 🐛 🐛 🐛

Affecter une expression à une variable déjà déclarée

#27 Fix 🐛 🐛 🐛

Utiliser la structure de contrôle while

#26 User story 🐛 🐛

Supporter les "escape character" dans les impressions
#25 <span>Fix</span> 
Demander au prof au suivi 2
#24 <span>Fix</span>
Vérifier Etape B Plus, Minus et UnaryMinus mal implémenté
#23 <span>Fix</span>   
Implémenter l'option -v de decac
#22 <span>Fix</span> 
Imprimer avec plusieurs paramètres
#20 <span>User story</span>  
Utiliser les structures de contrôle if, else
#19 <span>User story</span> 
Comparer des entiers et flottants
#18 <span>User story</span> 
Vérifier l'égalité des types int, float, boolean
#17 <span>User story</span> 
Faire des opérations arithmétiques sur les entiers et flottants
#16 <span>User story</span>  
Imprimer les variables de type int, float
#15 <span>User story</span>
Déclarer et instancier les types int, float, boolean
#14 <span>User story</span>  
Configurer les tests CI/CD et vérifier la non-régression
#7 <span>User story</span> 

## Sprint 3

Issues 37 Merge requests 53 Participants 15 Labels 8

Unstarted Issues (open and unassigned)

0

Ongoing Issues (open and assigned)

0

Completed Issues (closed)

37

Regler Assign pour Selection à gauche

#89 Priorité 1

Créer une classe vide

#51 Doing Priorité 1 User story

Implémenter Object.equals()

#56 Priorité 2 User story

Créer et appeler une méthode

#55 Priorité 2 User story

Créer et appeler un champ

#54 Priorité 2 User story

Déclarer et instancier une nouvelle classe

#53 Priorité 2 User story

Faire les casts avec les classes

#72 Priorité 3 User story

Appeler la classe avec this

#64 Priorité 3 User story

Executer du code assembleur

#62 Priorité 3 User story

Créer une méthode de classe avec une visibilité protected

#61 Priorité 3 User story

Créer une méthode prenant des paramètres

#60 Priorité 3 User story

Créer une méthode avec return
#59 <span>Priorité 3</span> <span>User story</span>  
Comparer l'instance d'une classe avec instanceof
#57 <span>Priorité 3</span> <span>User story</span>  
opti %2
#110
opti division / mul par puissance de 2
#109
Enlever test de débordement pour -opti
#108
correction convfloat return
#102 <span>Doing</span> <span>Fix</span>
Ajouter run-decompilation.py à mvn test
#99 <span>Technical Story</span> 
Ecrire le manuel utilisateur
#98 <span>User story</span>
Ajouter plus de tests
#96 <span>Technical Story</span>
Gérer New pour l'override avec type statique diff type dyna
#95 <span>Doing</span> <span>Fix</span>  
fix new Object
#93 <span>Fix</span> 
Supporter null
#90 <span>User story</span>



Methode override ne se trouve pas a la meme index dans la VTable
#83 <span>Fix</span> 
This provoque une erreur non détecté dans l'analyse contextuelle
#81 <span>Fix</span> 
Demander aux professeurs comment configurer un Job Gitlab
#77 <span>Technical Story</span>        
Construire le Control Flow Graph
#73 <span>OPTIM</span> <span>Technical Story</span> 
Améliorer TSTO
#71 <span>Technical Story</span> 
Optimisation : Retirer les CMP useless.
#70 <span>Fix</span>  
Lister des fonctionnalités spécifiques à Deca mal implémentés
#67 <span>Fix</span>
Améliorer la reconnaissance des commentaires par le lexer
#66 <span>Doing</span> <span>Fix</span>
Améliorer les tests de débordement
#65 <span>Fix</span> 
Documenter la conception de l'algorithme SSA
#63 <span>Doing</span> <span>Technical Story</span> 
Ajouter l'option -optim à decac
#58 <span>User story</span> 




Améliorer le codegen des expressions arithmétiques et booléennes

#52 Technical Story 

commencer un peu les classes

#46 Technical Story 

Faire expressions avec cast

#37 User story   

# Sprint 4

Issues 8 Merge requests 1 Participants 0 Labels 3

Unstarted Issues (open and unassigned)	Ongoing Issues (open and assigned)	Completed Issues (closed)
0	0	8
		Ecrire la documentation sur la gestion d'equipe et du projet #107 User story
		Ecrire la documentation sur l'analyse énergétique #106 User story
		Ecrire la documentation sur l'extension #105 User story
		Ecrire la documentation sur la validation #104 User story
		Ecrire la documentation de conception #103 User story
		Implémenter unSSA #76 OPTIM Technical Story
		Implémenter les algorithmes d'optimisation dans le SSA form #75 OPTIM Technical Story
		Convertir le CFG en SSA Form #74 OPTIM Technical Story