Bouton - Plus (+)Rapport Labo Claculatrice POO1

Tests GUI

Test	Voulu	Résussi
Bouton - Tous les chiffres	Affichage du caractère correspondant on chiffre presser dans la fenêtre	Oui
Bouton - Tous les chiffre: Lors d'erreur	Ne fait rien	Oui
Bouton - Chiffre (0): current value: 0	0	Oui
Bouton - Point : après 123	Ajout du point à la suite de la chaine de caractère courante: 123.	Oui
Bouton - Point (.): Après un chiffre qui contient déjà un point	Ne fait rien	Oui
Bouton - Point(.): après un Clear	0.	Oui
Bouton - Clear (C): Lorsque la console contient quelque chose dans "currentValue" + pile	Réinitialise l'affichage, supprime une eventuelle erreur et vide la pile	Oui
Bouton - Clear (C): Lorsqu'il y a rien dans la pile	Réinitialise l'affichage supprime une eventuelle erreur	Oui
Bouton - Clear (C): Lorsqu'il y a une erreur	Réinitialise l'affichage, supprime une eventuelle erreur et vide la pile	Oui
Bouton - ClearError (CE) : lorsqu'il y a une erreur	Réinitialise l'affichage à 0 et supprime d'eventuelle erreur	Oui
Bouton - ClearError (CE) : lorsqu'il y a pas d'erreur	Réinitialise l'affichage à 0	Oui
Bouton - ClearError (CE): lorsque quelque chose dans la stack	Réinitialise l'affichage à 0 mais conserve la stack	Oui
Bouton - Division(/): currentValue: "5.0" Stack:"2.0"	2.5	Oui
Bouton - Division(/): stack: "5.0"	Error	Oui
Bouton - Division(/): currentValue: "5.0" Stack:"0.0"	Erreur: No division by 0 !	Oui
Bouton - Division(/): lors d'erreur	Ne fait rien	Oui
Bouton - Enter	Ajout la currentValue sur la stack	Oui
Bouton - Enter: Lors d'erreur	Ne fait rien	Oui
Bouton - Inversed (1/x): currentValue "5.0"	0.2	Oui
Bouton - Inversed (1/x): currentValue "0.0"	Erreur	Oui

Test	Voulu	Résussi
Bouton - Inverse (1/x) : lors d'erreur	Ne fait rien	Oui
Bouton - MemoryRecall (MR): Il y a quelque chose de save dans la mémoire	Remplace la currentValue par l'élément sauvegardé	Oui
Bouton - MemoryRecall (MR): Il y a rien de save dans la mémoire	Ne fait rien	Oui
Bouton - MemoryRecall (MR): Lors d'erreur	Ne fait rien	Oui
Bouton - MemoryStore (MS): Quand rien n'a été sauvegarder avant	Sauvegarde l'élément	Oui
Bouton - MemoryStore (MS): Quand il y a déjà un élément sauvegarder	Remplace l'élément sauvegarder	Oui
Bouton - MemoryStore (MS): lors d'erreur	Ne fait rien	Oui
Bouton - Minus (-): CurrentValue: "5.0" stack:"2.0"	3.0	Oui
Bouton - Minus (-): currentValue: "5.0"	Erreur	Oui
Bouton - Minus (-): Lors d'erreur	Ne fait rien	Oui
Bouton - Multiply (*): currentValue: "5.0" stack:"2.0"	10.0	Oui
Bouton - Multiply (*): currentValue: "5.0"	Error	Oui
Bouton - Multiply (*): Lors d'erreur	Ne fait rien	Oui
Bouton - Plus (+): currentValue: "5.0" stack:"2.0"	7.0	Oui
Bouton - Plus (+): currentValue: "5.0"	Error	Oui
Bouton - Plus (+): Lors d'erreur	Ne fait rien	Oui
Bouton - Sign(+/-): currentValue: 123.4	currentValue: -123.4	Oui
Bouton - Sign(+/-): currentValue: -123.4	currentValue: 123.4	Oui
Bouton - Sign(+/-): Lors d'erreur	Ne fait rien	Oui
Bouton - Square (x^2): currentValue: "5.0"	25.0	Oui
Bouton - Square (x^2): CurrentValue: "-5.0"	25.0	Oui
Bouton - Square (x^2): Lors d'erreur	Ne fait rien	Oui
Bouton - SquareRoot (Sqrt): currentValue: "25.0"	5.0	Oui
Bouton - SquareRoot (Sqrt): currentValue: "-2.''	Error: No sqrt of negative values !	Oui

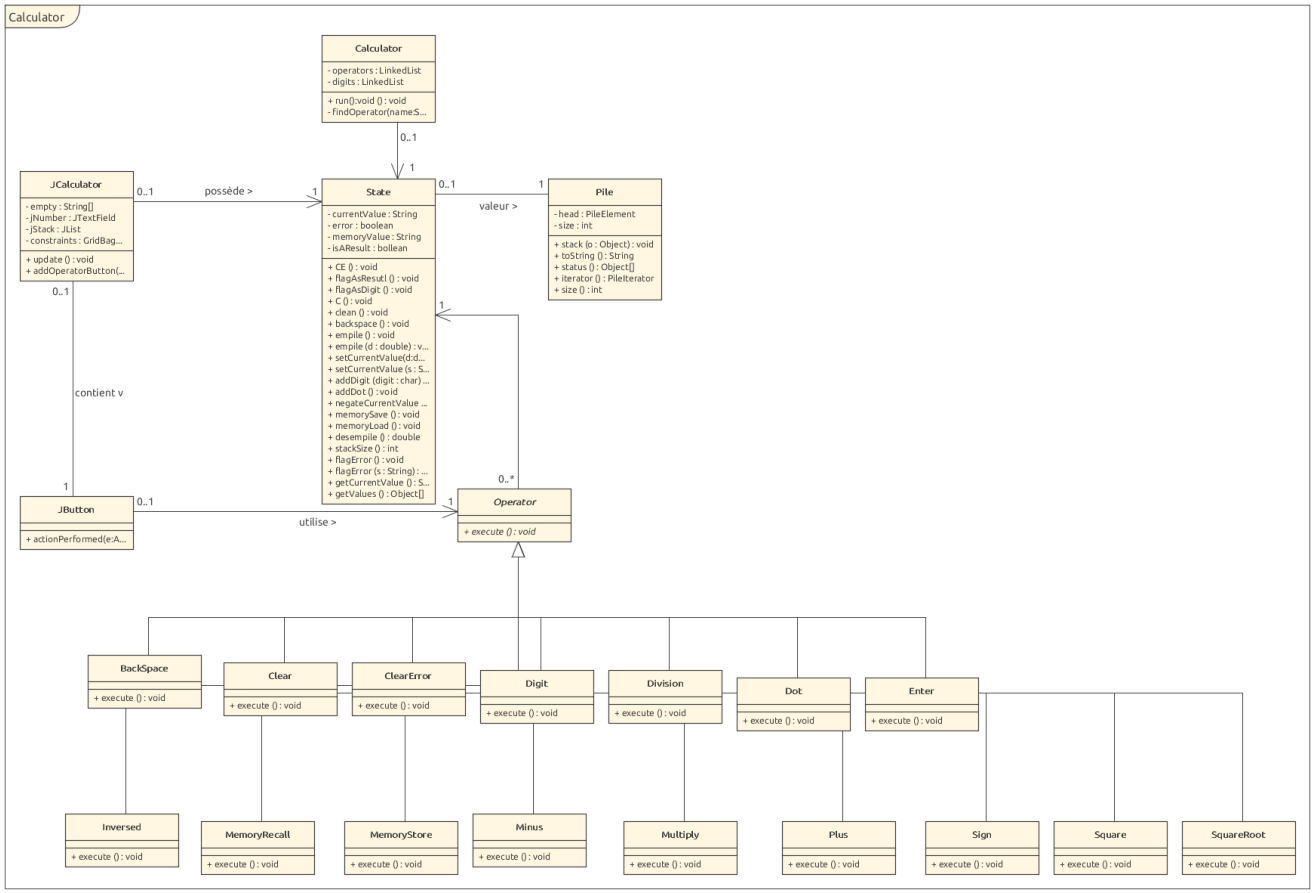
Test	Voulu	Résussi
Bouton - SquareRoot (Sqrt): Lors d'erreur	Ne fait rien	Oui
Bouton - BackSpace (<=): current value: 123.01	123.0	Oui
Bouton - BackSpace (<=): currentValue: 123.	123	Oui
Bouton - BackSpace(<=): currentValue: 0	0	Oui
Bouton - BackSpace (<=): currentValue: 1	0	Oui
Bouton - BackSpace (<=): Lors d'erreur	Ne fait rien	Oui

Test Console

Test	currentValue	Stack	Remarque	Réussi
1 <enter>	1	<>		Oui
2 <enter>	2	<1.0>		Oui
3 <enter>	3	<2.0> <1.0>		Oui
+ <enter>	5.0	<1.0>		Oui
- <enter>	-4.0	<>		Oui
+ <enter>	Error	<-4.0>	Error / on restack pas le - 4	Non
ce<enter>	0	<-4.0>		Oui
123.4 <enter>	123.4	<-4.0>		Oui
123hello.4 <enter>	123.4	<-4.0>	Ne fait rien	Oui
. <enter>	123.4	<-4.0>	Ne fait rien	Oui
.6 <enter>	0.6	<123.4> <-4.0>	A discuter/debug	Non
ms <enter>	0	<123.4> <-4.0>	saved: 0.6	Oui
5 <enter>	5	<123.4> <-4.0>	saved: 0.6	Oui
mr <enter>	0.6	<123.4> <-4.0>	saved: 0.6	Oui
+ <enter>	124.0	<-4.0>		Oui
mr <enter>	0.6	<-4.0>		Oui
0 <enter>	0	<0.6> <-4.0>		Oui
/ <enter>	No division by 0 !	<0.6> <-4.0>	Erreur	Oui
6 <enter>	No division by 0 !	<12.0> <0.0> <0.6> <-4.0>	Ne fait rien	Oui
ce <enter>	0	<12.0> <0.0> <0.6> <-4.0>	ClearError	Oui
c <enter>	0	<>	Clear	Oui
4 <enter>	4	<>		Oui
5 <enter>	5	<4.0>		Oui
* <enter>	20	<>		Oui

Test	currentValue	Stack	Remarque	Réussi
^ <enter>	400	<>		Oui
sqrt <enter>	20	<>		Oui
inv <enter>	0.05	<>		Oui
exit <enter>			Quitte le programme	Non

UML



Explication/Remarque

N'ayant pas vu le concepte de MVC, il nous a été difficile d'adapter correctement ce modèle. En fin de laboratoire, après discussion et reflexion avec nos collègue, nous pensons avoir fait une erreur de conception. Nous avons pensez que toutes les opérations accédant à la pile doit ce faire dans le *state*. Par concequence, les opérateurs ne font que appeler des fonctions dans la *state* qui réalise des opérations.

Si nous devions refaire ce laboratoire, nous ferions ces opérations dans les *opérateurs* et créer des fonctions qui permettent de faire des modifications de la pile sécurisées dans notre *state*.

Ayant vu les classes interne que en fin de laboratoire et suite à la demande de Pier de ne pas le faire, nous avons pas implémenter les opérateurs en tant que classe interne à *Operator*.

Conclusion

La difficulté fut de réalisé du code compatible avec la GUI et la console. Etant donner que en console, l'insertion de chiffre et l'utilisation de la touche *Enter* est différent conceptuellement.

L'utilisation d'un modèle comme un MVC est nouveau pour nous et ne sommes pas sur d'avoir complétement compris le concept.