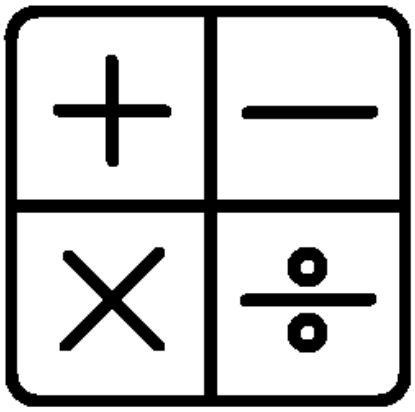


Seda

BrainUp inc.

Soutenance 1

Table des matières :

1. Introduction

2. Inspiration

3. Répartition des tâches

4. Tableau d'avancement

5. Fonctionnalitées calculatoires
    5.1 Shunting Yard
    5.2 Méthode de Newton-Rhapson
    5.3 Calcul d'expressions mathématiques
    5.4 Calcul dans différentes bases -soutenance 2-
    5.5 Opérations sur les matrices -soutenance 2-

6. Fonctionnalitées polynômiales
    6.1 Résolution de polynôme -soutenance 2-
    6.2 Traçage de courbe -soutenance 2-

7. Fonctionnalitées de conversion
    7.1 conversion physiques -soutenance 2-
    7.2 conversion fiduciaire -soutenance 2-

8. Fonctionnalitées mémoires
    8.1 sauvegarde de l'historique
    8.2 sauvegarde mémoire

9. Autre fonctionnalitées
    9.1 mini-jeux -soutenance 2-
    9.2 compilateur python

10. Site web

11. Conclusion

1. Introduction

Seda, ou calculatrice en coréen, est comme son nom l'indique un outil de calcul. Le projet Seda a pour objectif de créer une calculatrice capable de réaliser des opérations mathématiques de base, telles que les additions, les soustractions, les multiplications et les divisions.

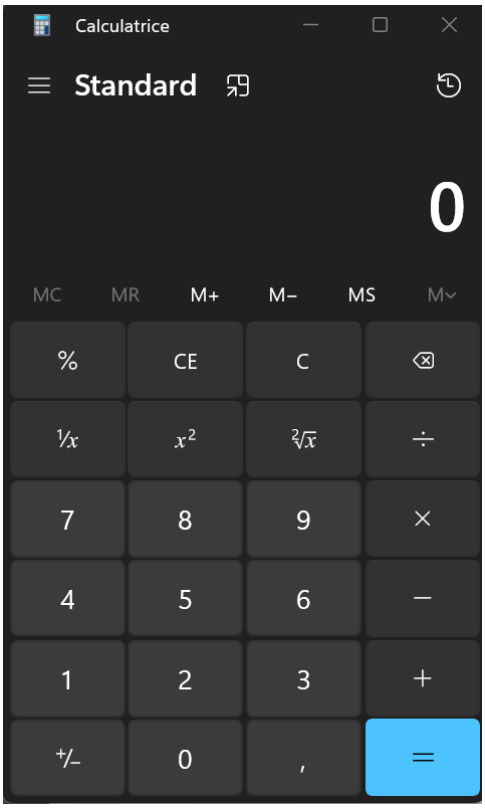
Cependant, le projet Seda ne se limite pas à ces fonctionnalités de base. En effet, il est également conçu pour calculer des fonctions mathématiques complexes, telles que la fonction exponentielle, la fonction logarithmique et d'autres fonctions mathématiques.

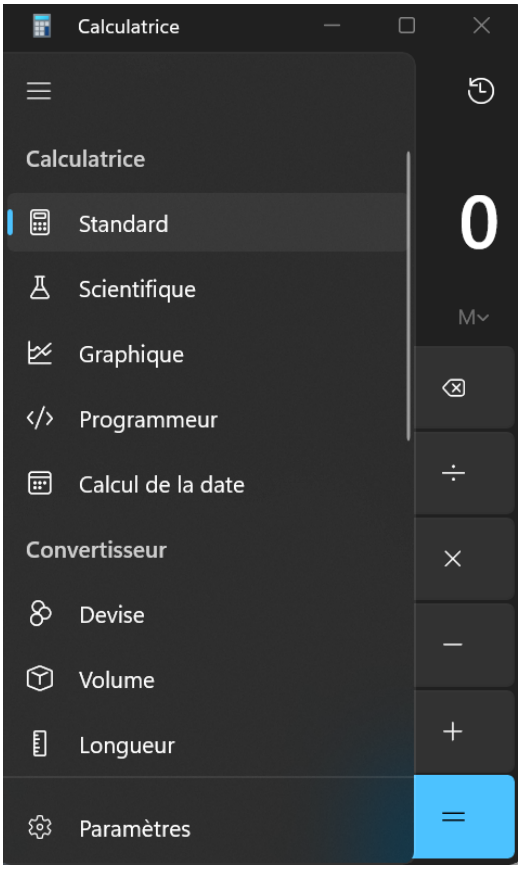
Le projet Seda va également plus loin en permettant de réaliser des calculs matriciels. Il sera possible de travailler avec des matrices et de les manipuler en utilisant les opérations matricielles de base. Par ailleurs, Seda pourra afficher les courbes associées à des polynômes, ce qui en fait un outil très pratique pour les mathématiques.

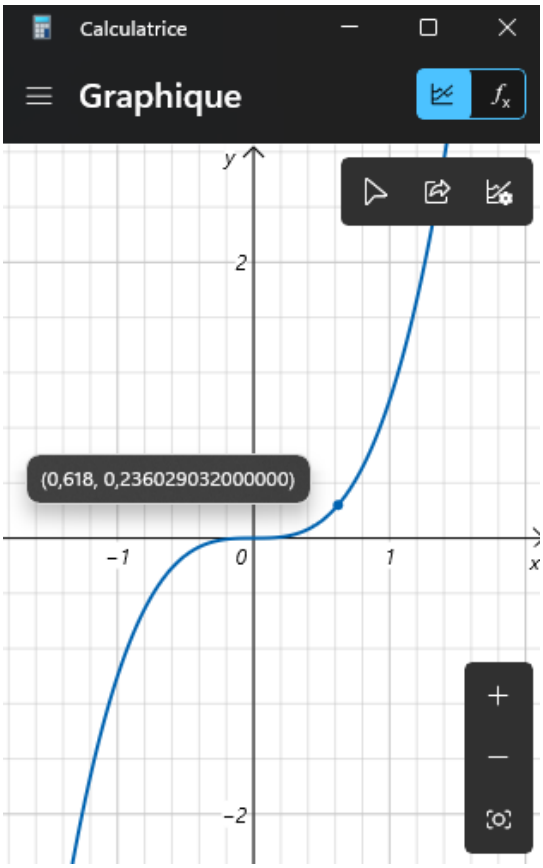
En résumé, le projet Seda vise à créer une calculatrice complète et performante qui pourra répondre aux besoins de différents utilisateurs. Avec ses nombreuses fonctionnalités, Seda deviendra un outil essentiel pour les étudiants en mathématiques, les chercheurs et les ingénieurs qui travaillent avec des données mathématiques complexes.

2. Inspiration

Notre inspiration pour ce projet Seda provient principalement de la calculatrice de base de Windows 11. Nous souhaitons recréer cet outil avec une attention particulière portée à la qualité et à la fiabilité des calculs. Notre ambition est de reproduire la plupart des fonctionnalités de la calculatrice de Windows 11, ce qui permettra aux utilisateurs de bénéficier d'un outil pratique et efficace pour réaliser des calculs simples ou plus complexes. Cependant, nous souhaitons également améliorer cette calculatrice en y ajoutant des fonctionnalités supplémentaires, telles que la possibilité de réaliser des calculs matriciels et d'afficher les courbes associées à des polynômes. L'objectif final de Seda est de devenir une calculatrice puissante et fiable, capable de répondre aux besoins de tout type d'utilisateur.




Calculatrice
Calculatrice
Standard
Scientifique
Graphique
Programmeur
Calcul de la date
Convertisseur
Devise
Volume
Longueur
Paramètres
0
M


Calculatrice
Graphique
y
2
(0,618, 0,236029032000000)
-1
0
1
x
-2

3. Répartition des tâches

	RHD	LED	GLM	MM
expressions mathématiques			✓	
calcul dans différentes bases	✓		✓	
opérations sur les matrices		✓	✓	✓
résolution de polynômes	✓			
traçage de courbe		✓		
conversions	✓			✓
gestion mémoire	✓			
ui		✓		
ux		✓		
site web				✓
mini-jeux	✓	✓	✓	✓
compilateur python	✓			

4. Tableau d'avancement

	Soutenance 1	Soutenance 2	Soutenance 3
expressions mathématiques	80%	100%	100%
calcul dans différentes bases	/	100%	100%
opérations sur les matrices	/	50%	100%
résolution de polynômes	/	50%	100%
traçage de courbe	/	30%	100%
conversions	/	50%	100%
gestion mémoire	60%	100%	100%
ui	70%	80%	100%
ux	70%	80%	100%
site web	80%	100%	100%
mini-jeux	/	20%	100%
compilateur python	20%	100%	100%

avancement atteint pour la Soutenance 1:

	Soutenance 1	Soutenance 2	Soutenance 3
expressions mathématiques	80%	100%	100%
calcul dans différentes bases	/	100%	100%
opérations sur les matrices	/	50%	100%
résolution de polynômes	/	50%	100%
traçage de courbe	/	30%	100%
conversions	/	50%	100%
gestion mémoire	80% (+20%)	100%	100%
ui	70%	80%	100%
ux	70%	80%	100%
site web	80%	100%	100%
mini-jeux	/	20%	100%
compilateur python	80% (+60%)	100%	100%

5. Fonctionnalités calculatoires

5. Implémentation de l'algorithme de calcul

Premièrement, on récupère une chaine de caractère qui a été créé à partir des de l'interface de la calculatrice. Ensuite, il faut convertir cette chaine de caractères en liste de Tokens afin de pouvoir traiter plus facilement chaque élément (opérateur, opérande, Parenthèse….). Il est possible de créer une liste de Token à l'aide de la struct suivante:

```
struct Token_Type
{
    struct Token_Operators* operator;
    struct Token_Operands* operands;
    struct Token_Parenthese* paranthese;
};
```

Si le token est un opérateur, alors l'attribut opérateur est différent de nulle, s'il le token est un opérande, alors l'attribut opérandes sera différent de null et finalement, si le token est une parenthèse, alors l'attribut parenthèse sera différent de null.

une fois notre liste de token créé, il nous faut quelque structure de données afin de réaliser l'algorithme de shunting yard: pile, file et éventuellement un arbre binaire pour évaluer le résultat.

Ensuite, on realise l'algorithme de shunting yard dans le but de pouvoir passer en RPN (notation polonaise inverse), puis une fois notre liste de token transforme en RPN, on peut alors construire un arbre binaire d'expression régulière puis il suffit de faire un parcours en profondeur en ordre infixe afin de pouvoir évaluer le résultat.

5.1 Shunting Yard

L'algorithme de Shunting Yard est un algorithme utilisé pour convertir une expression mathématique écrite en notation infixée (où les opérateurs sont entre les opérandes) en notation postfixée (où les opérateurs se trouvent après les opérandes). Il permet de convertir les expressions mathématiques de manière à les rendre plus faciles à évaluer par une machine. L'algorithme de Shunting Yard utilise deux piles : une pile d'opérandes et une pile

d'opérateurs. Il parcourt l'expression infixée caractère par caractère, et agit en fonction du type de caractère rencontrée :

Si le caractère est un opérande (comme un nombre ou une variable), il est ajouté à la pile d'opérandes. Si le caractère est un opérateur, il est ajouté à la pile d'opérateurs, à condition que les opérateurs déjà présents dans la pile aient une priorité inférieure ou égale à celle de l'opérateur en cours. Si un opérateur de priorité supérieure est présent, il est dépilé et ajouté à la pile d'opérandes avant d'ajouter l'opérateur en cours à la pile d'opérateurs. Si le caractère est un parenthese ouvrante, il est ajouté à la pile d'opérateurs. Si le caractère est un parenthese fermante, les opérateurs sont dépilés de la pile d'opérateurs et ajoutés à la pile d'opérandes jusqu'à ce que la parenthese ouvrante correspondante soit rencontrée. Une fois que l'expression infixée a été entièrement parcourue, les opérateurs restants dans la pile d'opérateurs sont dépilés et ajoutés à la pile d'opérandes. La pile d'opérandes contient alors l'expression postfixée, dans laquelle les opérateurs se trouvent après les opérandes. Il est important de noter que l'algorithme de Shunting Yard est utilisé pour évaluer les expressions mathématiques de manière à garantir la priorité des opérateurs, en respectant les règles de précédence des opérateurs mathématiques standard. Il permet également de gérer les parenthèses dans les expressions mathématiques de manière à garantir que les calculs sont effectués dans l'ordre correct. En résumé, l'algorithme de Shunting Yard est un algorithme utilisé pour convertir une expression mathématique écrite en notation infixée en notation postfixée, en utilisant deux piles : une pile d'opérandes et une pile d'opérateurs. Il garantit la priorité des opérateurs en respectant les règles de précédence des opérateurs mathématiques standard et gère les parenthèses pour assurer que les calculs sont effectués dans l'ordre correct. Cette notation postfixée est ensuite utilisée pour évaluer facilement l'expression mathématique par une machine.

5.2 Méthode de Newton-Rhapson

La méthode de Newton-Raphson est un moyen efficace pour trouver les racines d'une fonction mathématique. Elle est basée sur les principes de la dérivation, et utilise une itération pour se rapprocher de plus en plus de la solution. Pour utiliser la méthode de Newton-Raphson, vous devez d'abord choisir une valeur initiale pour x, appelée x0. Vous devez également choisir une précision pour la solution, appelée epsilon. Ensuite, vous calculez la fonction (x) et sa dérivée f'(x) en utilisant x0. La nouvelle valeur de x, appelée x1, est alors calculée en utilisant la formule suivante : x1 = x0 - f(x0) / f'(x0). Vous répétez ensuite ce processus en utilisant x1 à la place de x0, et en continuant à calculer x2, x3, etc. jusqu'à ce que

la différence entre xn et xn-1 soit inférieure à epsilon. La dernière valeur de x obtenue est considérée comme étant la racine de la fonction. Il est important de noter que la méthode de Newton-Raphson n'est pas toujours garantie de converger vers la solution, et que la valeur initiale x0 doit être choisie avec soin pour éviter les divergences. Il est également important de vérifier que f'(x) est non nulle, car la méthode nécessite de diviser par cette dérivée. En résumé, la méthode de Newton-Raphson est une méthode itérative pour trouver les racines d'une fonction, qui utilise les principes de la dérivation pour s'approcher progressivement de la solution. Elle nécessite un choix judicieux de la valeur initiale et de la précision, ainsi qu'une vérification de la non nullité de la dérivée de la fonction.

5.3 Calcul d’expressions mathématiques

La calculatrice pourra réaliser tout type d’opérations arithmétiques (additions, soustractions, multiplications, divisions, parenthèses, ln, exponentielle, modulo).

5.4 Calcul dans différentes bases -soutenance 2-

La calculatrice pourra également utiliser d’autres bases que la base 10. Ainsi l’utilisateur n’aura qu’à indiquer avant son calcul dans quelle base calculer. Le résultat pourra au choix être affiché dans la base du calcul ou sa valeur en base 10.

5.5 Opérations sur les matrices -soutenance 2-

La calculatrice sera également apte à réaliser différentes opérations matricielles telles que la multiplication par un scalaire, l’addition, la multiplication et la puissance.

6. Fonctionnalitées polynômiales

6.1 Résolution de polynôme -soutenance 2-

La calculatrice sera en mesure de trouver les racines d'un polynômes via le "root finding algorithm"

6.2 Traçage de courbe -soutenance 2-

Un bouton permettant l'affichage de courbes sera disponible, il sera alors possible de visualiser les diverses fonctions mathématiques usuelles (ln, sinus, cosinus…) ainsi que les combinaisons possibles entre celles-ci.

7. Fonctionnalitées de conversion

7.1 conversion physiques -soutenance 2-

Il sera possible de savoir le nombre de grammes dans X kilogrammes ou le nombre de litres dans X décilitres.

7.2 conversion fiduciaire -soutenance 2-

Il sera possible de savoir combien coûtent X dollars en euro au cours actuel (à l'heure) ou combien de centimes font X euros.

8. Fonctionnalitées mémoires

8.1 sauvegarde de l’historique

L’intégralité des calculs effectués et des résultats sont sauvegardés et peuvent être affichés à tout moment par l’utilisateur. Il suffit pour cela de cliquer sur le bouton “historique”, l’historique complet depuis la création de la calculatrice sera alors affiché dans le terminal de l’utilisateur.

8.2 sauvegarde mémoire

Il est possible de sauvegarder un résultat pour le réutiliser dans une opération plus tard. Il faut pour cela cliquer sur le bouton “M+”, le dernier résultat calculé sera alors enregistré. Cliquer sur le bouton “M-” permettra de récupérer cette valeur et de l’utiliser dans le calcul actuel.

9. Autre fonctionnalitées

9.1 mini-jeux -soutenance 2-

Différents jeux seront également implémentés, notamment un jeu de Tetris et un Pong.

9.2 compilateur python

Il est possible pour l'utilisateur d'écrire et voir le résultat de l'exécution d'un code python directement dans la calculatrice.
L'implémentation dans la calculatrice n'a pas encore été faite, le compilateur doit être utilisé à part. il faut pour cela écrire son programme python dans un fichier .txt, puis lancer le compilateur avec le fichier .txt en argument.
Le résultat de la compilation sera alors affiché dans la terminal de l'utilisateur.

Programme python "python_test_file.txt" pour tester le compilateur python

```
1  print(3+8)
2  print()
3  print('ceci est un texte')
4  print()
5  for i in range(4):
6          print('ceci est un texte puis un nombre:',i)
```

résultat de l'exécution de ce programme dans un terminal

```
◆ BrainUp_Corp. git:(main) ❯❯❯ ./python_executable python_test_file.txt
11

ceci est un texte

ceci est un texte puis un nombre: 0
ceci est un texte puis un nombre: 1
ceci est un texte puis un nombre: 2
ceci est un texte puis un nombre: 3
```

10. Site web

Concernant le site web, celui-ci à été entièrement implémenté en Javascript, CSS ainsi que HTML. Celui-ci est accessible à l'adresse suivante : Matthiaskd.github.io et est scindé en plusieurs parties. On peut alors y retrouver la section ‘’what is Seda ‘’ qui décrit simplement et globalement l'application. Vient ensuite la section ‘’features’’ qui sera mise à jour tout au long du développement de ce projet. En effet, chaque partie accomplie aura un manuel d'utilisation sur le site. La page web présentera des versions simplifiées ‘’démos’’ des fonctionnalités implémentées. Pour finir, en bas de page se trouve la section ‘’membres’’ qui présente simplement l'équipe du projet ainsi que leurs différentes responsabilités.

11. Conclusion