AMYOT Flavie
SALIBA Louise
CB2

Rapport TB7 : Partie Interprétation du code généré

Le but de cette partie du projet est de récupérer l'équation placée dans un arbre binaire créé par ceux en charge de l'analyse Syntaxique (Jérémy et Bastien), afin de créer une fonction qui calcule l'image d'un x donné.
Nous avons dû faire une fonction qui étudiait tous les cas les uns après les autres (type et valeur de l'élément dans le noeud). Pour cela nous avons utilisé les fonctions "switch" et "case : ", afin d'éviter les successions de if.

Pour se faire, nous avions commencé par faire une fonction qui retournait un double : la valeur de f(x). Or nous avons rencontré un problème concernant les erreurs sur une opération (division par zéro, racine négative, etc..), nous n'avions pas de moyen de retourner un cas d'erreur en indiquant son type. Nous avons donc créé une structure Result qui comporte un type d'erreurs (no_erreur s'il n'y en a pas, le type de l'erreur si il y a une erreur), et qui prend également la valeur de f(x) s'il n'y a pas d'erreur, 0 sinon. Ainsi, Mathis et Marianne qui s'occupent de la partie graphique, peuvent récupérer les structures retournées par notre fonction et vérifier s'il y a ou non une erreur avant de les placer dans un tableau pour pouvoir tracer la courbe.

Nous avons utilisé la récursivité de façon à ce que lorsque l'on rencontre un opérateur, le calcul effectué soit " (résultat fils droit) opérateur (résultat fils gauche) ", ainsi nous n'avions plus qu'à identifier l'opérateur et réaliser l'opération.
De plus, nous nous sommes mis d'accord avec Bastien et Jérémy pour que lorsque l'utilisateur fait appel à une fonction (exemple : sinus), le fils de la fonction soit forcément à gauche. En effet, une fonction a forcément uniquement un seul fils (qui lui peut en avoir plusieurs). De cette façon, nous évitons de devoir dédoubler notre code ou de devoir créer une autre fonction pour afin de voir si le contenu de la fonction est à droite ou à gauche.

Afin de pouvoir gérer les erreurs mathématiques, nous avons défini plusieurs types d'erreurs. Nous avons donc défini ces erreurs pour les différentes fonctions. L'erreur de division par zéro pour les divisions, sinus, cosinus et tangentes cardinaux, et l'erreur d'opération non réelle pour les racines carrée de nombres négatifs, et les ln ou log de nombres négatifs ou nuls.

Une fois notre programme terminé, nous l'avons testé de toutes les façons possibles : l'intégralité des opérations et fonctions de notre répertoire. De plus, nous avons fait attention à vérifier le fonctionnement de toutes les erreurs possibles : valeurs interdites et divisions par 0.

Le problème possible que l'utilisateur pourrait rencontrer serait que nous n'avons pas traité le cas ou il souhaite calculer un nombre négatif à la puissance 1/nombre impair. Bien qu'impossible, ce cas n'afficherait pas de message d'erreur, sans toutefois donner de résultat.