UNIVERSITÉ LAVAL
DÉPARTEMENT D'INFORMATIQUE ET DE GÉNIE LOGICIEL

Devoir # 2 - Automne 2021
IFT-3101
Compilation et interprétation

Devoir à remettre au plus tard le vendredi 19 novembre 2021 à 23h59.

Ce devoir a pour but de vous faire écrire certaines parties d'un compilateur.

Ce devoir est à faire en équipe de trois obligatoirement.

1 Énoncé

Dans ce travail, vous devrez modifier le compilateur de *C$* afin de lui ajouter certaines fonctionnalités. Les fonctionnalités à ajouter sont les suivantes

— Énoncé `break` et `continue` sans étiquette
— Énoncé `break` et `continue` avec étiquette
— Transtypage des opérateurs arithmétiques et de comparaison

Les sections suivantes vous donnent quelques informations supplémentaires pour chacune des parties.

2 Énoncé `break` et `continue`

Modifiez le compilateur *C$* pour implémenter les énoncés `break` et `continue` sans étiquette. Voici la description de l'énoncé `break` sans étiquette :

> Un énoncé `break` transfère le contrôle à l'extérieur du bloc d'énoncé englobant.
>
> *BreakStatement* : `break`
>
> Un énoncé `break` sans étiquette essaie de transférer le contrôle au plus proche bloc d'énoncé `switch`, `while`, `do`, `repeat` ou `for` ; cet énoncé, appelé cible d'arrêt se termine donc normalement. Pour être précis, un énoncé `break` sans étiquette termine toujours abruptement l'énoncé englobant. Si aucun énoncé `switch`, `while`, `do`, `repeat` ou `for` n'est présent dans le bloc englobant, une erreur de compilation est levée.

Voici également la description de l'énoncé continue :

> Un énoncé `continue` peut apparaître seulement dans un énoncé `while`, `do`, `repeat` ou `for` ; un énoncé de cette sorte est appelé énoncé d'itération. Le contrôle se transfère au point de continuation de la boucle d'un énoncé d'itération.
>
> *ContinueStatement* : `continue`
>
> Un énoncé `continue` sans étiquette essaie de transférer le contrôle au plus proche énoncé `while`, `do`, `repeat` ou `for` englobant ; cet

énoncé, appelé la cible du `continue`, termine immédiatement l'itération courante et en recommence une nouvelle. Si aucun `while`, `do`, `repeat` ou `for` n'englobe l'énoncé `continue`, une erreur de compilation est levée.

2.1 Énoncé `break` et `continue` avec étiquette

Dans le langage *C$*, il devrait être possible de nommer un énoncé de boucle. Un `break` ou `continue` devrait donc pouvoir spécifier la boucle englobante à arrêter.

Une boucle nommée a la structure suivante :

LoopStatement : *loop* `as` *LoopName*

LoopName est le nom associé à la boucle. Un `break` ou un `continue` avec étiquette n'est valide que si le nom de l'étiquette est le nom associé à l'une des boucles des portées englobants l'énoncé `break` ou `continue`. Le comportement de l'énoncé est le même que pour celui sans étiquette, à l'exception de la cible qui est celle de l'itération associée à l'étiquette. Si le nom de l'étiquette n'est pas celui d'une boucle englobante, une erreur de compilation est levée.

3 Transtypage

Modifiez le compilateur *C$* pour implémenter le transtypage au niveau des expressions dans lesquelles apparaissent des opérateurs arithmétiques (binaires) et des opérateurs de comparaison.

Présentement les opérateurs (binaires) du compilateur *C$* sont homogènes, c'est-à-dire qu'ils n'acceptent que deux opérandes de même type. Ainsi, si i et j sont des entiers (`int32`) et si e et f sont des réels en virgule flottante (`float32`), les expressions $i + j$ et $e + f$ sont compilées correctement. Cependant les expressions $i + f$ et $e + j$ génèrent des erreurs de typage.

Il s'agit donc pour cet exercice de permettre les opérations binaires mixtes, c'est-à-dire les opérations arithmétiques (+, -, \, * et %) et les opérations de comparaison (==, !=, <, <=, >, >=), selon la spécification donnée dans le tableau suivant. Du code est déjà présent de le compilateur pour vous aider à commencer.

Opérandes		Version de	Coercition d'opérande	
gauche	droite	l'opérateur	gauche	droite
`int`	`int`	*int*		
`float`	`float`	*float*		
`int`	`float`	*float*	`int` à `float`	
`float`	`int`	*float*		`int` à `float`

4 Soumission

Vous devez soumettre un fichier nommé `compilateur.zip` contenant l'ensemble des fichiers du compilateur. Vous devez aussi fournir un fichier test contenant des tests pour l'ensemble des fonctionnalités que vous avez ajouté. Le fichier test doit se nommer `test.ccash`.

Bonne chance !