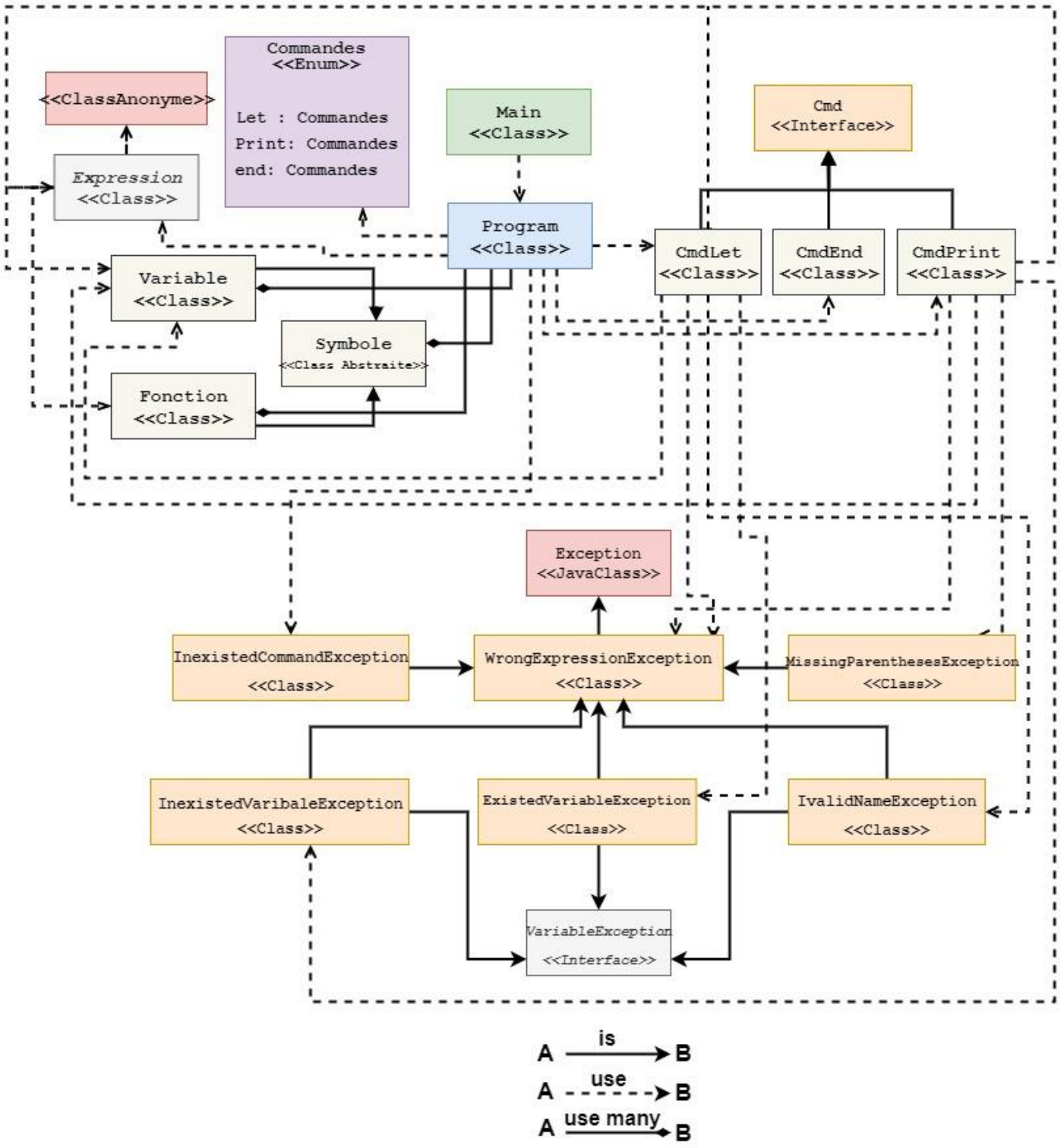
# ***La Partie Conception TP POO***

## 1-Diagramme Des Classes :



# 2-Tableaux Des Classes :

## Public Class Program

### *Liste des attributs*

| *Attribut* | *Signification* |
|---|---|
| `Public static ArrayList<Symbole> symbols` | Garder les symboles (Variables ou Fonction) avec ses informations. |
| `Public static Expression exp` | Garder l'expression correspondant à la commande saisie par l'utilisateur |

### *Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public static void addStandardsFunctions()` | Charger les fonctions standards dans le tableau symboles. |
| `public static void launch () throws InexistedCommandException` | Lancer le programme et écouter et traiter les commandes saisies. |

## Public Class Expression

### *Liste des attributs*

| *Attribut* | *Signification* |
|---|---|
| `Private String exp` | Garder la valeur de l'expression. |

### *Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public void Expression(String exp)` | Créer une instance expression avec un valeur spécifie. |
| `public double evaluate()` | Vérifier et valider la syntaxe de l'expression et retourner le résultat si elle est valide. |
| `public void replaceVariables() throws InexistedVariableException` | Vérifier l'existence des variables de l'expression et les remplacer par leur valeur trouvée dans symbols. |
| `Public void evalParentheses() throws MissingParenthesesException` | Vérifier les parenthèses dans l'expression cad à chaque p fermante il y a p ouvrante. |
| `Public void setExp(String exp)` | Instancier l'attribut exp. |
| `Public String getExp()` | Retourner la valeur de exp |

## Public Abstract Class Symbole

***Liste des attributs***

| *Attribut* | *Signification* |
|---|---|
| `Private String nom` | Garder le nom du symbole. |

***Liste des méthodes***

| *Method* | *Rôle* |
|---|---|
| `public Symbole()` | Créer un symbole vide. |
| `public Symbole(String nom)` | Créer un symbole avec un nom spécifie. |
| `public abstract int recherche()` | Chercher un symbole dans le tableau symbols et retourner son indice. |
| `Public void setNom(String nom)` | Instancier l'attribut nom. |
| `Public String getNom()` | Retourner la valeur de nom. |

## Public Class Variable extends Symbole

***Liste des attributs***

| *Attribut* | *Signification* |
|---|---|
| `Private Double value` | Garder la valeur de la variable. |

***Liste des méthodes***

| *Method* | *Rôle* |
|---|---|
| `public Variable()` | Créer une variable vide. |
| `public Variable(String nom, Double value)` | Créer une variable avec un nom et un value spécifies. |
| `public void ajouter()` | Ajouter la variable dans le tableau symbols. |
| `Public int recherche()` | Chercher une variable dans le tableau symbols et retourner son indice. |
| `Public String getValue()` | Retourner la valeur de value. |
| `Public void setValue(Double value)` | Instancier l'attribut value. |

## Public Class Fonction extends Symbole

***Liste des attributs***

| *Attribut* | *Signification* |
|---|---|
| `Private Function corps` | Garder le corps de la fonction du symbole. |

***Liste des méthodes***

| ***Method*** | ***Rôle*** |
|---|---|
| `public Fonction()` | Créer une Fonction vide. |
| `public Fonction(String nom, Function<Double, Double> corps)` | Créer une Fonction avec un nom et corps spécifies. |
| `public abstract int recherche()` | Chercher une fonction dans le tableau symbols et retourner l'indice. |
| `Public void setCorps(Function<Double, Double> corps)` | Instancier l'attribut corps. |
| `Public Function<Double, Double> getCorps()` | Retourner la valeur de corps. |

## Interface Cmd

***Liste des méthodes***

| ***Method*** | ***Rôle*** |
|---|---|
| `Public void execute()` | Exécuter la commande correspondante. |

## Public Class CmdLet implements Cmd

***Liste des attributs***

| ***Attribut*** | ***Signification*** |
|---|---|
| `Private static Stack<String> stk` | Garder les variables trouvées dans l'expression saisie. |

***Liste des méthodes***

| ***Method*** | ***Rôle*** |
|---|---|
| `public void execute()` | (Redéfinition de la méthode défini déjà dans l'interface cmd), Créer la variable et affecter sa valeur puis l'ajouter au tableau symbols s'il est valide |
| `public static void processVaraible(String nom)` | Vérifier et valider la syntaxe de variable et l'ajouter au symbols s'il est valide. |

## Public Class CmdPrint implements Cmd

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public void execute()` | (Redéfinition de la méthode défini déjà dans l'interface cmd), Calculer et Afficher le résultat de la commande après la validation |

## Public Class CmdEnd implements Cmd

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public void execute()` | (Redéfinition de la méthode défini déjà dans l'interface cmd), Vider le tableau symbols, arrêter le programme et afficher un message d'arrêt. |

## Public Class WrongExpressionException extends Exception

*Liste des attributs*

| *Attribut* | *Signification* |
|---|---|
| `Private String errorText` | Garder le message d'erreur de l'exception. |

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public MissingExpressionException()` | Créer une Exception vide. |
| `public MissingExpressionException(String errorText)` | Créer une Exception un message d'erreur . |
| `Public void setErrorText(String errorText)` | Instancier l'attribut errorText. |
| `Public String getErrorText()` | Retourner la valeur de errorText. |

## Public Class MissingParenthesesException extends WrongExpressionException

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public MissingParenthesesException()` | Créer une Exception vide. |
| `Public MissingParenthesesException (String errorText)` | Créer une Exception un message d'erreur . |

## Public Class InexistedCommandException extends WrongExpressionException

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public InexistedCommandException()` | Créer une Exception vide. |
| `Public InexistedCommandException(String errorText)` | Créer une Exception un message d'erreur. |

## Interface VariableException

Pour séparer les exceptions concernant les variables pour mieux organiser

## Public Class InexistedVariableException implements VariableException

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public InexistedVariableException()` | Créer une Exception vide. |
| `Public InexistedVariableException(String errorText)` | Créer une Exception un message d'erreur. |

## Public Class ExistedVariableException extends WrongExpressionException implements VariableException

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public ExistedVariableException()` | Créer une Exception vide. |
| `Public ExistedVariableException(String errorText)` | Créer une Exception un message d'erreur. |

## Public Class InvalidVariableNameException implements VariableException

*Liste des méthodes*

| *Method* | *Rôle* |
|---|---|
| `public InvalidVariableNameException()` | Créer une Exception vide. |

| | |
|---|---|
| `Public InvalidVariableNameException(String errorText)` | Créer une Exception un message d'erreur. |

## Enum Commandes

***Liste des commandes***

| ***Method*** | ***Rôle*** |
|---|---|
| `let` | Initialiser les variables. |
| `print` | Afficher le résultat sur console |
| `end` | Terminer le programme |

## Classe anonyme utilisé dans la méthode evaluate de Classe Expression

***Liste des attributs***

| ***Attribut*** | ***Signification*** |
|---|---|
| `Private int pos` | Serve comme indexe de parcours de l'expression. |
| `Private int ch` | Serve comme un conteneur de char dans l'expression |

***Liste des méthodes***

| ***Method*** | ***Rôle*** |
|---|---|
| `public void nextChar()` | Mettre le char prochain dans la variable ch. |
| `public boolean isExist(int ch)` | Vérifier si ch existe dans l'expression. |
| `public double parseTerm()` | Evaluer les expressions écrites comme <facteur> [ [ "*" \| "/" ] <facteur> ]... |
| `Public double parseFactor()` | Evaluer les expressions écrites comme <element> [ "^" <element> ]... |
| `Public double parseExpression()` | Evaluer les expressions écrites comme [ "-" ] <term> [ [ "+" \| "-" ] <term> ]... |
| `Public double parse()` | Evaluer et retourner le résultat de toute l'expression donnée |