HM Langage

Les fonctions :

```
type func identif(type identif, ...){
    instructions;
    return identif;
}
```

Les procédures :

```
func identif(type identif, ...){
    instructions;
}
```

Appel fonction :

```
var identif(type);
identif = fct(arguments);
```

Appel procédure :

```
fct(arguments);
```

Déclaration Variable :

1. Sans initialisation :

```
var identif(type);
```

2. Avec initialisation :

```
var identif(type) = valeur;
```

Remarque :

Pour éviter le problème d'initialisation des variables, Si nous avons utilisé une variable non initialisée, elle prend une valeur par default selon le type de cette dernière (number : 0 , string : NULL, Boolean: true).

if / elif / else :

```
if(conditions){
        instructions;
}
elif(conditions){
        instructions;
}
else{
        instructions;
}
```

Boucle for :

```
For identif from valeur_initiale – valeur_finale {
        instructions;
}
```

Boucle while :

```
While(conditions){
        instructions;
}
```

Boucle do / while :

```
do{
        instructions;
} while(conditions);
```

Afficher :

1. Afficher une phrase :

```
        out(" phrase ");
```

2. Afficher une variable :

```
        out(" |identif|  ");
```

3. Afficher plusieurs variables :

```
out(" |identif|   |identif1| ");
```

4. Afficher une phrase avec variables :

```
out( " phrase |identif| ");
```

Lire :

```
in(identif);
```

Types :

Number (entier ou reel)

String (chaine de caracteres)

Boolean (true ou false)

Commentaire :

```
// paragraphe
```