# コンパイラ資料（変数機能と記号表）　　担当：佐々木晃

## 本日の話題（１）

原始言語 → 翻訳（コンパイル）→ 目的言語

□演習で用いる原始言語(source language)

□目的言語(hsm 仮想機械) (target language)

□翻訳方法(compile / translation)

## □演習で用いる原始言語の説明（前回資料）

### ○C. 変数の機能

前回資料

## □hsm（スタックマシン型 CPU）の機械語への翻訳の方法（つづき）

### ○変数機能の実現(前回例の再掲)

(1) ソースプログラム

```
int main(){
   int a, b;
   a = 123;
   b = 789 + a;
}
```

○(2) 逆ポーランド

```
123 a :=
789 a + b :=
```

○(3) 擬似 hsm コード

```
DECL a b
LDC 0 123   (123)
STV 0 a     (a :=)
LDC 0 789   (789)
LDV 0 a     (a)
AD  0 0     (+)
STV 0 b     (b :=)
```

○(4) hsm コード

```
PUSH 0 2
LDC 0 123   (123)
STV 0 0     (a := )
LDC 0 789   (789)
LDV 0 0     (a)
AD   0 0    (+)
STV 0 1     (b := )
POP 0 2
```

### □記号表

```
int main(){
   int a, b;
   a = 123;
   b = 789 + a;
}
```

int a, b;のように a,b 二つの変数が宣言されたとする。このとき、
スタックの底から数えて 0 番目の位置（0 番地）に a の値を置き、
スタックの底から数えて 1 番目の位置（1 番地）に b の値を置く
と決めることにしよう。これを記号表として（名前表、シンボルテーブルなどとも）まとめると、

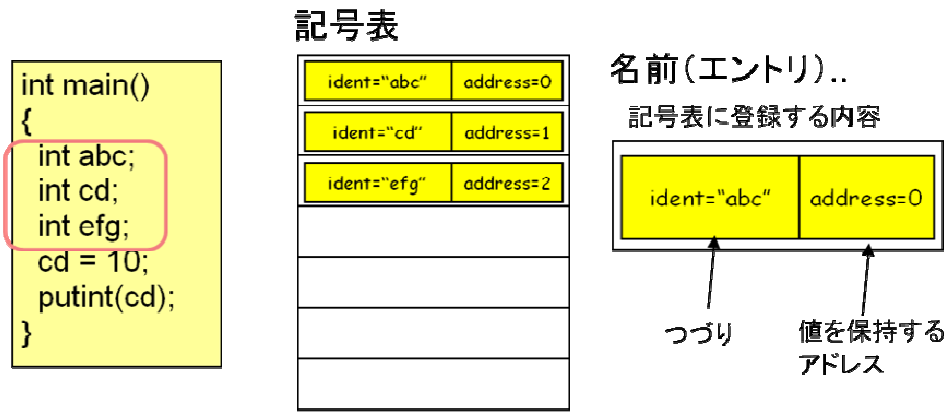
| つづり | 番地 |
|---|---|
| a | 0 |
| b | 1 |

となる。これは、変数から番地への写像（Map）となっている。表を見ることで、
「変数 a の値を置くと決めた番地」=0
「変数 b の値を置くと決めた番地」=1
とすぐに知ることができる。番地は、宣言した順番に割り当てればよい。（他の順番でも別によい。）

### □記号表を用いた変数名の解決(基本的アイディア)

○宣言部：表に順番に名前を登録していく。このときにスタック上のどの位置にそれぞれの変数を割り付けるかを決定する。

○文(statement)中での使用：変数が出てくる(出現、occurrence)のは、代入文の左辺および、式中での変数参照である。今、変数 x が出てきたとする。x に割り付けたアドレスを取り出すには、
(1) 記号表を検索し、つづり部が x であるエントリを取り出す。
(2) そのエントリのアドレス部を取り出す。




## □変数宣言および使用におけるエラー（意味エラー）の検出

### ○変数の二重定義エラーのチェック

○「宣言部」では、表に順番に名前を登録していくが、ある変数 x が途中で宣言されたときに、すでに同じつづりを持つエントリが存在していないかをチェックする。もし存在すれば、

「変数 x はすでに定義されています。」

"redefinition of variable x"

のようなエラーを出す。

### ○未定義の変数使用のチェック

**○文(statement)中での使用：変数が出てくる(出現、occurrence)のは、代入文の左辺および、式中での変数参照である。今、変数 x が出てきたとする。x に割り付けたアドレスを取り出すには、**

**(1) 記号表を検索し、つづり部が x であるエントリを取り出す。**

**(2) そのエントリのアドレス部を取り出す。**

と書いたが、(1)の際に「つづり部が x であるエントリを取り出そうとしたが」見つからない状態は、すなわちその変数 x が宣言部で宣言されていないということである。

## □記号表（まとめ）

○宣言部：表に順番に名前を登録していくが、ある変数 x が途中で宣言されたときに、すでに同じつづりを持つエントリが存在していないかをチェックする。

もし存在すれば、

「変数 x はすでに定義されています。」

"redefinition of variable x"

のようなエラーを出す。

○文(statement)中での使用：変数が出てくる(出現、occurrence)のは、代入文の左辺および、式中での変数参照である。今、変数 x が出てきたとする。x に割り付けたアドレスを取り出すには、

(1) 記号表を検索し、つづり部が x であるエントリが存在するかチェックする。

→存在しなければ未定義変数の使用としてエラーを発生させる。

「変数 x は宣言されていません。」
( variable x not declared)
→存在すれば、
(2)つづり部が x であるエントリを取り出し、
(3) そのエントリのアドレス部を取り出す。