TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

PHÂN TÍCH NGỮ NGHĨA

QUẢN LÝ PHẠM VI

Nhóm 05

Danh sách thành viên

STT	Họ và tên	Phụ trách
1	Nguyễn Văn Đạt	checkFreshIdent, checkDeclaredIdent
2	Nguyễn Trọng Cảnh	checkDeclaredConstant, checkDeclaredType
3	Vũ Long Vũ	checkDeclaredVariable, checkDeclaredFunction
4	Lưu Thị Thanh Ngân	checkDeclaredProcedure, checkDeclaredValueIdent
5	Bùi Thị Kim Anh	lookUpObject

Nội dung

- Kiểm tra sự trùng lặp khi khai báo biên đối tượng
- Kiểm tra tham chiếu tới các đối tượng

Nội dung

- Kiểm tra tên hợp lệ
- Kiểm tra định danh đã khai báo
- Kiểm tra hằng số đã khai báo
- Kiểm tra kiểu đã khai báo
- Kiểm tra biến đã khai báo
- Kiểm tra hàm đã khai báo
- Kiểm tra thủ tục đã khai báo

Nhiệm vụ

1. checkFreshIdent
2. checkDeclaredIdent
3. checkDeclaredConstant
4. checkDeclaredType
5. checkDeclaredVariable
6. checkDeclaredProcedure
7. checkDeclaredFunction
8. checkDeclaredLValueIdent

Các mã lỗi

- ERR_UNDECLARED_IDENT
- ERR_UNDECLARED_CONSTANT
- ERR_UNDECLARED_TYPE
- ERR_UNDECLARED_VARIABLE
- ERR_UNDECLARED_FUNCTION
- ERR_UNDECLARED_PROCEDURE
- ERR_DUPLICATE_IDENT

checkFreshIdent

- Mục đích: kiểm tra tên hợp lệ
- Được sử dụng khi:
  - Khai báo hằng
  - Khai báo kiểu người dùng định nghĩa
  - Khai báo biến
  - Khai báo tham số hình thức
  - Khai báo hàm
  - Khai báo thủ tục

```
void checkFreshIdent(char *name) {
    // TODO
    Object * obj = findObject(symtab->currentScope->objList, name);
    if (obj != NULL)
        error(ERR_DUPLICATE_IDENT, currentToken->lineNo, currentToken->colNo);
}
```

checkDeclaredIdent

- Mục đích: kiểm tra định danh đã được định nghĩa
- Được sử dụng khi factor là một ident

```
Object* checkDeclaredIdent(char* name) {
  // TODO
  // Start search
  on_search = 0;

  Object *obj = lookupObject(name);

  // Reset search
  on_search = 0;

  if (obj == NULL)
      error(ERR_UNDECLARED_IDENT, currentToken->lineNo, currentToken->colNo);

  return obj;
}
```

checkDeclaredConstant

Mục đích: Kiểm tra hằng số được khai báo
Được sử dụng khi khai báo hằng, đảm bảo hằng số có giá trị

```
Object* checkDeclaredConstant(char* name) {
  Object *obj = NULL;
  on_search = 0;
  do {
      obj = lookupObject(name);
      if (obj != NULL && obj->kind == OBJ_CONSTANT)
          break;
  } while (obj != NULL);
  on_search = 0;
  if (obj == NULL)
      error(ERR_UNDECLARED_CONSTANT, currentToken->lineNo, currentToken->colNo);
  return obj;
}
```

checkDeclaredType

Mục đích: Kiểm tra kiểu khai báo

Được dùng để kiểm tra xem kiểu khai báo có tồn tại hay không?

```
Object* checkDeclaredType(char* name) {
  Object *obj = NULL;
  on_search = 0;
  do {
      obj = lookupObject(name);
      if (obj != NULL && obj->kind == OBJ_TYPE)
          break;
  } while (obj != NULL);
  on_search = 0;
  if (obj == NULL)
      error(ERR_UNDECLARED_TYPE, currentToken->lineNo, currentToken->colNo);
  return obj;
}
```

checkDeclaredVariable

- Mục đích: kiểm tra biến được khai báo
  - Được sử dụng khi định danh là biến

```
114  Object* checkDeclaredVariable(char* name) {
115      Object *obj = NULL;
116
117      // Start search
118      on_search = 0;
119
120      do {
121          obj = lookupObject(name);
122          if (obj != NULL && obj->kind == OBJ_VARIABLE)
123              break;
124      } while (obj != NULL);
125
126      // End search
127      on_search = 0;
128
129      if (obj == NULL)
130          error(ERR_UNDECLARED_VARIABLE, currentToken->lineNo, currentToken->colNo);
131
132      return obj;
133  }
```

checkDeclaredFunction

- Mục đích: kiểm tra hàm được khai báo

```
135  Object* checkDeclaredFunction(char* name) {
136      Object *obj = NULL;
137
138      // Start search
139      on_search = 0;
140
141      do {
142          obj = lookupObject(name);
143          if (obj != NULL && obj->kind == OBJ_FUNCTION)
144              break;
145      } while (obj != NULL);
146
147      // End search
148      on_search = 0;
149
150      if (obj == NULL)
151              error(ERR_UNDECLARED_FUNCTION, currentToken->lineNo, currentToken->colNo);
152
153      return obj;
154  }
```

checkDeclaredProcedure

- Mục đích: kiểm tra thủ tục khai báo
- Sử dụng khi: có tham chiếu tới thủ tục đó (lệnh gọi)

```
Object* checkDeclaredProcedure(char* name) {
  // TODO
  Object *obj = NULL;

  // Start search
  on_search = 0;

  do {
      obj = lookupObject(name);
      if (obj != NULL && obj->kind == OBJ_PROCEDURE)
          break;
  } while (obj != NULL);

  // End search
  on_search = 0;

  if (obj == NULL)
          error(ERR_UNDECLARED_PROCEDURE, currentToken->lineNo, currentToken->colNo);

  return obj;
}
```

checkDeclaredValueIdent

- Mục đích: Dùng để kiểm tra object có phải là hàm, biến hoặc tham số

```
Object* checkDeclaredLValueIdent(char* name) {
  // TODO
  Object *obj = NULL;

  // Start search
  on_search = 0;

  do {
      obj = lookupObject(name);
      if (obj != NULL && (obj->kind == OBJ_FUNCTION || obj->kind == OBJ_PARAMETER || obj->kind == OBJ_VARIABLE))
          break;
  } while (obj != NULL);

  // End search
  on_search = 0;

  if (obj == NULL)
          error(ERR_UNDECLARED_IDENT, currentToken->lineNo, currentToken->colNo);

  return obj;
}
```

Trả lời câu hỏi

1. **Mô tả hoạt động hàm LookupObject**

- Khởi tạo currentScope bằng null để lưu trữ phạm vi hiện tại của object
- Bắt đầu tìm kiếm ở phạm vi hiện tại (currentScope) không thực hiện tìm kiếm ở thời điểm trước đó hoặc một đối tượng toàn cục đã được thực hiện (đối tượng toàn cục như WRITEI, WRITEC, WRITELN, READC, READI.)
- Thực hiện tìm kiếm trong ObjList (những đối tượng cùng thuộc phạm vi) sau đó chuyển currentScope sang phạm vi bên ngoài (outer liên kết với phạm vi bên ngoài)
- Nếu tìm thấy thì đánh dấu object đã được tìm kiếm =>Trả về đối tượng tìm thấy
- Sau đó nếu không tìm thấy trong danh sách cùng phạm vi thì sẽ tìm đến danh sách các dối tượng toàn cục (vẫn tìm kiếm trong phạm vi hiện tại curentScope)
- Sau đó đánh dấu là đã kết thúc quá trình tìm kiếm
- Trả về đối tượng

Trả lời câu hỏi

2. Những đối tượng nào có thể sử dụng mà không cần khai báo? Chúng có thuộc cùng phạm vi với chương trình chính hay không?

- Những đối tượng toàn cục là WRITEI, WRITEC, WRITELN, READC, READI.
- Chúng không thuộc cùng phạm vi với chương trình chính

=> vì theo như việc tìm kiếm đối tượng thông qua phạm vi thì họ thực hiện tìm kiếm trong danh sách các đối tượng cùng phạm vi (ObjList) rồi mới tìm kiếm trong danh sách đối tượng phạm vi rộng hơn và cuối cùng là danh sách các đối tượng toàn cục