Started on	Saturday, 24 June 2023, 8:08 PM
State	Finished
Completed on	Saturday, 24 June 2023, 10:08 PM
Time taken	2 hours
Marks	1.00/5.00
Grade	**2.00** out of 10.00 (**20**%)

Question 1

Not answered

Mark 0.00 out of 1.00

Given the description of a program in BKOOL as follows:

A program in BKOOL consists of many declarations, which are **variable** and **function declarations**.

Modify the BKOOL.g4 as follows:

program: // write for program rule here using vardecl and funcdecl

vardecl: 'vardecl' ;

funcdecl: 'funcdecl' ;

WS: [\t\r\n] -> skip;

ERROR_CHAR: . {raise ErrorToken(self.text)};

For example:

Test	Result
"""vardecl"""	successful

Answer: (penalty regime: 0 %)

```
1
```

Question 2

Not answered

Mark 0.00 out of 1.00

Given the description of a program in BKOOL as follows:

A program in BKOOL consists of many declarations, which are **variable** and **function declarations**.

A **variable declaration** starts with a type, which is **int** or **float**, then a comma-separated list of identifiers and ends with a semicolon.

A **function declaration** also start with a type and then an identifier, which is the function name, and then parameter declaration and ends with a body. The parameter declaration starts with a left round bracket '(' and a null-able semicolon-separated list of parameters and ends with a right round bracket ')'. Each parameter always starts with a type and then a comma-separated list of identifier.

Modify BKOOL.g4 as follows:

program: // write your rule here

//And some other rules for variable declaration, function declaration and other rules

body: 'body';

ID: // includes a sequence of alphabetic characters.

WS: [\t\r\n] -> skip;

ERROR_CHAR: . {raise ErrorToken(self.text)};

For example:

Test	Result
"""int a, b,c; float foo(int a; float c, d) body float goo (float a, b) body"""	successful

Answer: (penalty regime: 0 %)

```
1
```

Question 3

Not answered

Mark 0.00 out of 1.00

Given the description of a program in BKOOL as follows:

A program in BKOOL consists of many declarations, which are **variable** and **function declarations**.

A **variable declaration** starts with a type, which is **int** or **float**, then a comma-separated list of identifiers and ends with a semicolon.

A **function declaration** also start with a type and then an identifier, which is the function name, and then parameter declaration and ends with a body. The parameter declaration starts with a left round bracket '(' and a null-able semicolon-separated list of parameters and ends with a right round bracket ')'. Each parameter always starts with a type and then a comma-separated list of identifier. A body starts with a left curly bracket '{', follows by a null-able list of variable declarations or statements and ends with a right curly bracket '}'.

There are **3 kinds of statements**: assignment, call and return. All statements must end with a semicolon. An assignment statement starts with an identifier, then an equal '=', then an expression. A call starts with an identifier and then follows by a null-able comma-separated list of expressions enclosed by round brackets. A return statement starts with a symbol 'return' and then an expression.

Modify BKOOL.g4 as follows:

program :// write your rule for program here

//And some other rules for variable declaration, function declaration, statements but using following expr for an expression

expr: 'expr';

ID: //includes a sequence of alphabetic characters

WS: [\t\r\n] -> skip;

ERROR_CHAR: . {raise ErrorToken(self.text)};

For example:

Test	Result

"""int a, b,c;
float foo(int a; float c, d) {
  int e ;
  e = expr ;
  c = expr ;
  foo(expr);
  return expr;
}
float goo (float a, b) {
  return expr;
}"""
	successful

Answer: (penalty regime: 0 %)

```
1
```

Question 4

Correct

Mark 1.00 out of 1.00

Given the description of a program in BKOOL as follows:

A program in BKOOL consists of many declarations, which are **variable** and **function declarations**.

A **variable declaration** starts with a type, which is **int** or **float**, then a comma-separated list of identifiers and ends with a semicolon.

A **function declaration** also start with a type and then an identifier, which is the function name, and then parameter declaration and ends with a body. The parameter declaration starts with a left round bracket '(' and a null-able semicolon-separated list of parameters and ends with a right round bracket ')'. Each parameter always starts with a type and then a comma-separated list of identifier. A body starts with a left curly bracket '{', follows by a null-able list of variable declarations or statements and ends with a right curly bracket '}'.

There are **3 kinds of statements**: assignment, call and return. All statements must end with a semicolon. An assignment statement starts with an identifier, then an equal '=', then an expression. A call starts with an identifier and then follows by a null-able comma-separated list of expressions enclosed by round brackets. A return statement starts with a symbol 'return' and then an expression.

An **expression** is a construct which is made up of operators and operands. They calculate on their operands and return new value. There are four kinds of infix operators: '+', '-', '*' and '/' where '+' have lower precedence than '-' while '*' and '/' have the highest precedence among these operators. The '+' operator is right associative, '-' is non-associative while '*' and '/' is left-associative. To change the precedence, a sub-expression is enclosed in round brackets. The operands can be an integer literal, float literal, an identifier, a call or a sub-expression.

For example:

```
int a, b, c;
float foo(int a; float c, d) {
     int e;
     e = a + 4;
     c = a * d / 2.0;
     return c + 1;
}
float goo(float a, b) {
     return foo(1, a, b);
}
```

Some tokens:

1. An identifier includes a sequence of alphabetic characters.

2. An integer number includes a sequence of numerical characters.

3. A real (float) number includes two parts: integer and fractional parts. The integer and fractional part are like a integer number, but separated by a point (.).

Your task:

Write a grammar of the program in BKOOL using ANTLR and submit its generation files to this exercise.

For example:

Test	Result

"""int a, b,c;
float foo(int a; float c, d) {
  int e ;
  e = a + 4 ;
  c = a * d / 2.0 ;
  return c + 1;
}
float goo (float a, b) {
  return foo(1, a, b);
}"""
	successful

Answer: (penalty regime: 0 %)

BKOOL.interp

BKOOL.tokens

BKOOLLexer.interp

BKOOLLexer.py

BKOOLLexer.tokens

BKOOLParser.py

BKOOLVisitor.py

	Test	Expected	Got	

✔	"""int a, b,c;
	float foo(int a; float c, d) {
	int e ;
	e = a + 4 ;
	c = a * d / 2.0 ;
	return c + 1;
	}
	float goo (float a, b) {
	return foo(1, a, b);
	}"""
		successful	successful	✔

✔	"""int a, b,;"""	Error on line 1 col 9: ;	Error on line 1 col 9: ;	✔
✔	"""float foo(int a, float c, d) {}"""	Error on line 1 col 17: float	Error on line 1 col 17: float	✔
✔	"""float foo(int a; float c, d;) {}"""	Error on line 1 col 28:)	Error on line 1 col 28:)	✔
✔	"""int c; c = 4;"""	Error on line 2 col 0: c	Error on line 2 col 0: c	✔

✔	"""int a, b,c;
	float foo(int a; float c, d) {
	int e = 5;
	e = a + 4 ;
	c = a * d / 2.0 ;
	return c + 1;
	}
	float goo(float a, b) {
	return foo(1, a, b);
	}"""
		Error on line 3 col 9: =	Error on line 3 col 9: =	✔

	Test	Expected	Got	

✔	"""int a, b,c;
	float foo(int a; float c, d) {
	int e ;
	e = a + 4 ;
	c = a * d / 2.0 ;
	return c + 1
	}
	float goo (float a, b) {
	return foo(1, a, b);
	}"""
		Error on line 7 col 0: }	Error on line 7 col 0: }	✔

✔	"""float goo (float a, b) {
	return foo(1, a, b);
	}
	
	c = 5;"""
		Error on line 5 col 0: c	Error on line 5 col 0: c	✔

✔	"""float goo (float a, b) {
	return foo(1, a, b) + 1;
	}"""
		successful	successful	✔

✔	"""float goo (float a, b) {
	return 1 - foo(1, a, b);
	}"""
		successful	successful	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question 5

Not answered

Mark 0.00 out of 1.00

Trong ngôn ngữ **MiniPHP**, chương trình bao gồm các khai báo biến. Một *khai báo biến* được gắn liền với lần đầu tiên biến đó được gán giá trị. Phép gán trong MiniPHP bao gồm các thành phần theo thứ tự tên biến ***VARNAME***, dấu bằng ***EQ***, một biểu thức và kết thúc bởi dấu chấm phẩy ***SEMI***.

Biểu thức trong MiniPHP là tổ hợp của các *toán hạng* và các *toán tử* được viết theo trung thứ tự (infix expression).

- Các toán hạng bao gồm: tên biến, hằng số nguyên ***INTLIT***, hằng số thực ***FLOATLIT***, hằng chuỗi ***STRINGLIT*** hoặc một mảng. Có hai loại mảng trong MiniPHP là mảng chỉ số (indexed array) và mảng phối hợp tên (associative array).

+ *Mảng chỉ số* bắt đầu bằng từ khóa array **ARRAY** tiếp theo là một danh sách có thể rỗng các biểu thức được phân cách bởi một dấu phẩy ***COMMA*** và được bao lại bằng một cặp ngoặc tròn ***LP*** và **RP**.

+ *Mảng phối hợp tên* bằng từ khóa array **ARRAY** tiếp theo là một danh sách có thể rỗng *các cặp kết hợp* (associative pair) được phân cách bởi một dấu phẩy ***COMMA*** và được bao lại bằng một cặp ngoặc tròn ***LP*** và ***RP***. Một cặp kết hợp bao gồm một tên cặp ***PAIRNAME***, tiếp theo là dấu mũi tên ***ARROW*** và sau đó là một biểu thức.

- Các toán tử được liệt kê theo độ ưu tiên từ cao xuống thấp (các toán tử được mô tả trên cùng một dòng sẽ cùng một độ ưu tiên) và chỉ rõ tính kết hợp:

+ Toán tử ** ***DSTAR***: kết hợp phải

+ Toán tử . **DOT**: kết hợp trái

+ Toán tử * ***MUL***, / ***DIV***, % ***MOD***: kết hợp trái

+ Toán tử + ***ADD***, - ***SUB***: kết hợp phải.

+ Toán tử ?? ***DQUES***: không có tính kết hợp.

- Để thay đổi được độ ưu tiên và tính kết hợp, người ta có thể sử dụng cặp ngoặc tròn để tạo biểu thức con.

Sử dụng ANTLR4 để mô tả ngôn ngữ nói trên. Biết rằng các từ in đậm và nghiêng là tên các từ vựng trong ngôn ngữ đã đặt, sinh viên sử dụng dạng thức đơn giản nhất để mô tả.

Ví dụ:

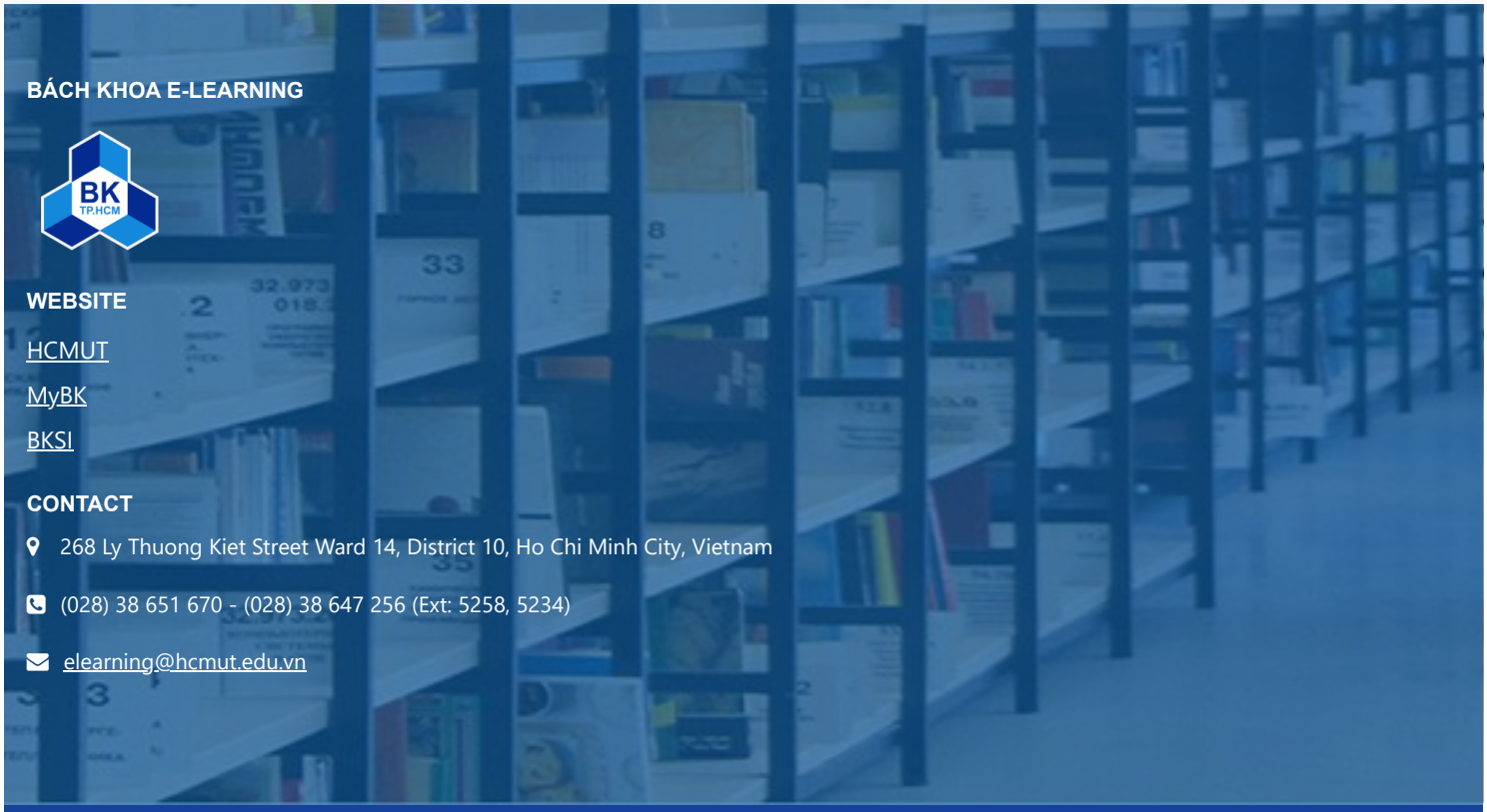
```
abc = 1 + 2 ?? 3;
u = array(a1 => 3 . 4, a2 => 3 + (u2 % 5));
```

For example:

Test	Result
"""abc = 1 + 2 ?? 3;"""	successful

Answer: (penalty regime: 10, 20, ... %)

```
1
```



BÁCH KHOA E-LEARNING

WEBSITE

HCMUT

MyBK

BKSI

CONTACT

268 Ly Thuong Kiet Street Ward 14, District 10, Ho Chi Minh City, Vietnam

(028) 38 651 670 - (028) 38 647 256 (Ext: 5258, 5234)

elearning@hcmut.edu.vn