コンパイラ・期末テスト問題用紙
（2024 年 02 月 13 日・ 10:30 ～ 12:00）

解答上、その他の注意事項

1. 問題は、問 Ⅰ～Ⅲ までである。

2. 解答用紙の右上の欄に学籍番号・名前を記入すること。

3. 解答欄を間違えないよう注意すること。

4. 解答中の文字 (特に a と d)がはっきりと区別できるよう注意すること。

5. 持ち込みは不可である。筆記用具・時計・学生証以外のものは、かばんの中などにしまうこと。

6. テストの配点は 70 点である。合格は毎週の課題の得点を加点して、100 点満点中 60 点以上とする。

I.（演算子順位法）

次の BNF で表される文法を演算子順位法により構文解析する。

$$E \rightarrow \quad \mathbf{id} \ \mid \ E \texttt{ == } E \ \mid \ E \texttt{ \&\& } E \ \mid \ E \texttt{ << } E \ \mid \ E \texttt{ @@ } E \ \mid \ \texttt{(}\ E\ \texttt{)}$$

ただし、「E」 は非終端記号、「**id**」,「==」,「&&」,「<<」,「@@」,「(」,「)」 は終端記号で、「**id**」 はアルファベット 1 文字からなるトークンを表す。

この文法は曖昧なので、優先順位と結合性について次のように決めておく。

「==」は非結合、「&&」は右結合、「<<」は左結合、「@@」は右結合、であり、「==」は「&&」よりも優先順位が高く、「&&」は「<<」よりも優先順位が高く、「<<」は「@@」よりも優先順位が高いものとする。

つまり、下表中の左の欄の式は、右の欄の式として解釈される。

式	解釈
a == b == c	（構文エラー）
a && b && c	a && (b && c)
a << b << c	(a << b) << c
a @@ b @@ c	a @@ (b @@ c)
a == b && c	(a == b) && c
a && b == c	a && (b == c)
a == b << c	(a == b) << c
a << b == c	a << (b == c)

式	解釈
a == b @@ c	(a == b) @@ c
a @@ b == c	a @@ (b == c)
a && b << c	(a && b) << c
a << b && c	a << (b && c)
a && b @@ c	(a && b) @@ c
a @@ b && c	a @@ (b && c)
a << b @@ c	(a << b) @@ c
a @@ b << c	a @@ (b << c)

以下の演算子順位行列の空欄 (1) ～ (13) を ⋖、≒、⋗、✗ のうちもっとも適切なもので埋めよ。ただし ✗ はエラーを表すものとする。（教科書などの記法では、エラーは空欄のままとしているが、このテストでは無回答と区別するために明示的に ✗ を書くことにする。）

左＼右	@@	<<	&&	==	(	)	id	終
始	⋖	⋖	⋖	⋖	⋖	✗	⋖	≒
@@	(1)	⋖	⋖	(2)	(3)	⋗	⋖	⋗
<<	⋗	(4)	(5)	⋖	⋖	(6)	⋖	⋗
&&	(7)	⋗	(8)	⋖	⋖	⋗	⋖	⋗
==	⋗	⋗	(9)	(10)	⋖	⋗	(11)	⋗
(	⋖	(12)	⋖	⋖	⋖	≒	⋖	✗
)	⋗	⋗	⋗	⋗	✗	⋗	✗	⋗
id	(13)	⋗	⋗	⋗	✗	⋗	✗	⋗

II.（再帰下降構文解析）

対のような BNF で定義された文法に対して再帰下降構文解析ルーチンを作成する。

$$
\begin{array}{lcl}
E & \to & \mathbf{id} \mid \texttt{\{}\, E \,\texttt{!}\, L \,\texttt{\}} \mid E\,\texttt{(}\,E\,\texttt{)} \\
L & \to & L\,\texttt{,}\,S \mid S \\
S & \to & \mathbf{id}\,\texttt{:}\,E \qquad\qquad \cdots \text{Ⓐ}
\end{array}
$$

ただし、「E」,「L」,「S」は非終端記号、「**id**」,「{」,「!」,「}」,「(」,「)」,「,」,「:」は終端記号である。このうち、「**id**」はアルファベット 1 文字からなるトークンを表す。 開始記号（start symbol）は E である。

(1) E から導出される終端記号の列で、次の条件を満たすものの例を挙げよ。存在しなければ ✗ を記せ。

(ⅰ)「**id**」の直後に「,」が続く。
(ⅱ)「:」の直後に「{」が続く。
(ⅲ)「!」の直後に「(」が続く。
(ⅳ)「)」の直後に「**id**」が続く。

(2) E, L から左再帰を除去せよ。補助的に導入する非終端記号はそれぞれ E', L' とせよ。後の解答で使用するために、生成規則に丸数字（①, ②, …）を付けておくこと。なお、採点の都合上、順番は次のようにせよ。

- 左再帰の除去で追加される ε は**最後の**選択肢とすること
- ε 以外の BNF の右辺の選択肢の順は、もとの BNF の選択肢の順と同じにすること
- 連続した番号をつけること

以下の問は (2). で E, L から左再帰を除去して得られた BNF について答えよ。

(3) $First(L)$ を求めよ。

(4) $Follow(L')$ を求めよ。

(5) $Follow(E')$ を求めよ。

(6) 下の予測型構文解析表の S の行を埋めよ。この問題の解答は ✗, Ⓐ の中から選べ。ただし、✗ は“エラー”を示す。 解答なしと区別するために、構文エラーの場合は、必ず ✗ を記入し、空欄のまま残さないこと。

(7) 下の予測型構文解析表の E, E', L, L' の行を埋めよ。この問題の解答は ✗ と ①, ②, …（(2) の解答で、BNF の生成規則に自分で付けた番号）から選べ。構文エラーの場合は、必ず ✗ を記入し、空欄のまま残さないこと。

	id	{	!	}	(	)	,	:	$
$E \to$									
$E' \to$									
$L \to$									
$L' \to$									
$S \to$									

(8) この文法に対して、入力が文法にしたがっていれば「正しい構文です。」間違っていれば「構文に誤りがあります。」と表示する構文解析プログラムを作成する。プログラム（次ページ）中の指定の部分に入る `E`, `E1`, `L`, `L1`, `S` 関数のうち、 `L`, `L1` 関数の定義を完成させよ。ただし、`E`, `E1`, `L`, `L1`, `S` は、それぞれ非終端記号 E, E', L, L', S に対応する関数

である。終端記号を消費するときは eat 関数を、予測型構文解析表の ✗ に相当する入力には reportError 関数を呼び出すようにすること。

プログラムの補足説明:

プログラム中では、終端記号は「!」のような 1 文字のものは、その字そのもの（の文字コード）、**id** などのトークンは、C 言語のマクロ（例えば **id** の場合は ID）として表現している。入力の終わり (\$) に対応するのは、このプログラムの場合、改行文字 '\n' である。

E, E1, L, L1, S 関数が実行されるときは token という大域変数に、現在処理中（入力の先頭）の終端記号が代入されている。 eat 関数は、現在 token に入っている値が、引数として与えられた終端記号と等しいかどうか確かめ、等しければ次の終端記号を token に読み込む。reportError 関数は、「構文に誤りがあります。」と表示し、プログラムを終了する。

再帰下降構文解析プログラム

```c
#include <stdio.h>   /* printf(), EOF など */
#include <stdlib.h>  /* exit()用     */
#include <ctype.h>   /* isalpha()用 */

/* 終端記号に対するマクロの定義 */
#define ID    257    /* トークン id */

int token;           /* 大域変数の宣言 */

/* 関数プロトタイプ宣言 */
void reportError(void);
void eat(int t);
int  yylex(void);

void E(void);
void E1(void);
void L(void);
void L1(void);
void S(void);

/* ******************************************************* */
/* この部分に 関数 E, E1, L, L1, S の定義を挿入する。          */
/* ******************************************************* */

/* ここ以降は解答に直接関係はない。 */
void eat(int t) { /* token を消費して、次の tokenを読む */
  if (token == t) {
    /* 現在のトークンを捨てて、次のトークンを読む */
    token = yylex();
    return;
  } else {
    reportError();
  }
}

void reportError(void) {
  printf("構文に誤りがあります。\n");
  exit(0); /* プログラムを終了 */
}
```

```
40
41 int yylex(void) {     /* 簡易字句解析ルーチン */
42   int c;
43   do { /* 空白は読み飛ばす。 */
44     c = getchar();
45   } while (c == ' ' || c == '\t');
46
47   if (c == EOF) { /* ファイルの終わり */
48     exit(0);
49   } else if (isalpha(c)) {  /* アルファベットだったら… */
50     return ID;
51   } else {
52     /* 上のどの条件にも合わなければ、文字をそのまま返す。*/
53     return c;   /* ';' など */
54   }
55 }
56
57 void processLine(void) {  /* 各行の処理 */
58   L();
59   if (token == '\n') {
60     /* 入力がブロックしないように改行は特別扱い */
61     printf("Correct!\n");   /* eat('\n') の前に出力しておく */
62   }
63   eat('\n');
64 }
65
66 int main(void) {   /* main関数 */
67   printf("Ctrl-c で終了します。\n");
68   token = yylex(); /* 最初のトークンを読む */
69   while (1) { /* 無限ループ */
70     processLine();  /* 各行を処理する */
71   }
72
73   return 0;
74 }
```

III.（LR 構文解析）

次のような BNF から LR 構文解析表を作成する。

$$
\begin{array}{lcll}
S & \to & \mathbf{x} & \cdots \text{ I} \\
 & | & B\ \texttt{\}} & \cdots \text{ II} \\
B & \to & B\,S & \cdots \text{ III} \\
 & | & \texttt{\{} & \cdots \text{ IV}
\end{array}
$$

ただし、

- ··· の後の Ⅰ, Ⅱなどは生成規則の番号である。
- 「S」,「B」は非終端記号である。「**x**」,「`{`」,「`}`」は終端記号である。このうち、「**x**」はアルファベット 1 文字からなるトークンを表す。
- 開始記号（start symbol）は S である。

`bison` の出力する LR 構文解析表は次のようになる。（**注:** `bison` に `-v` オプションを指定することによって、LR 構文解析表をファイルに出力させることができる。）

	x	{	}	$		S	B
⓪	s ①	s ②				g ③	g ④
①	r Ⅰ						
②	r Ⅳ						
③				s ⑤			
④	s ①	s ②	s ⑥			g ⑦	g ④
⑤	accept						
⑥	r Ⅱ						
⑦	r Ⅲ						

注: ここで、 s ㉑ は、「シフト (shift) して状態 ㉑ へ遷移」、 g ㉓ は、「状態 ㉓ へ遷移 (go)」、 r Ⅻ は、「生成規則 Ⅻ を使って還元 (reduce)」を表す。

オートマトンの開始状態は ⓪ である。

次の (1) ～ (4) の入力列に対して、**<u>下線の記号</u>をシフトした直後**の（つまりシフトしたあと、還元がまだ起こっていないときの）スタックの状態はどのようになっているか？

(1) { x <u>x</u> }　　(2) { { { } <u>x</u> } }

(3) { { } x <u>x</u> }　　(4) { x { x } { <u>x</u> } }

下の選択肢（(1) ～ (4) 共通）から選べ。（左がスタックの底とする。）

(A). ⓪B④**x**①　(B). ⓪{②B④**x**①　(C). ⓪B④B④**x**①　(D). ⓪{②B④{②B④**x**①

(E). ⓪B④S⑦**x**①　(F). ⓪{②B④S⑦**x**①　(G). ⓪B④B④}⑥**x**①

(H). ⓪B④{②B④}⑥**x**①　(I). ⓪B④B④B④}⑥**x**①　(J). ⓪{②B④{②B④{②B④}⑥**x**①

(K). ⓪B④B④S⑦B④}⑥**x**①　(L). ⓪{②B④{②B④S⑦{②B④}⑥**x**①

計算用紙

計算用紙

コンパイラ・期末テスト解答用紙（2024 年 02 月 13 日）

学籍番号		氏名	

I. (1 × 13)

(1)		(2)		(3)		(4)		(5)	
(6)		(7)		(8)		(9)		(10)	
(11)		(12)		(13)					

II. (2 × 4, 6, 3, 3, 3, 2, 8, 4)

(1)	(i)		(ii)	
	(iii)		(iv)	

(2)

$E \quad \rightarrow$

$E' \quad \rightarrow$

$L \quad \rightarrow$

$L' \quad \rightarrow$

(3) { }

(4) { }

(5) { }

(6)

	id	{	!	}	(	)	,	:	**$**
$S \rightarrow$									

(7)

	id	{	!	}	(	)	,	:	**$**
$E \rightarrow$									
$E' \rightarrow$									
$L \rightarrow$									
$L' \rightarrow$									

(8)	

III. (5 × 4)

(1)		(2)		(3)		(4)	

授業・テストの感想