記号処理・中野・火 4 講時

2015 年 8 月 4 日

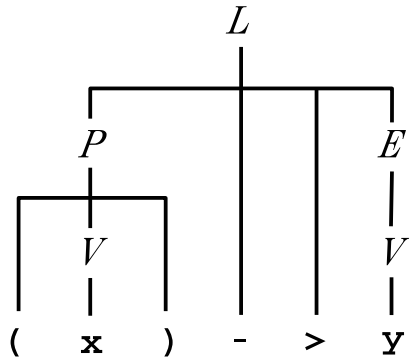
I　BNF 記法で書かれた次のような文法について以下の問に答えなさい。

$\langle V \rangle$ ::= “x” | “y” | “z” .
$\langle A \rangle$ ::= “+” | “-” .
$\langle P \rangle$ ::= $\langle V \rangle$ | “(” [$\langle V \rangle$ { “,” $\langle V \rangle$ }] “)” .
$\langle E \rangle$ ::= [$\langle A \rangle$] $\langle V \rangle$ | $\langle E \rangle$ $\langle A \rangle$ $\langle V \rangle$.
$\langle L \rangle$ ::= $\langle P \rangle$ “-” “>” $\langle E \rangle$.

(1) 終端記号列 x、x+y、(y,z)、()->x-y、x->yz、z->-z-z-z が、非終端記号 P、E、L から導出できるかどうかについて、次の表の空欄に、導出できる場合は○を、できない場合は×をそれぞれ書き込みなさい。(10 点)

終端記号列	P	E	L
x	○	○	×
x+y	×	○	×
(y,z)	○	×	×
()->x-y	×	×	○
x->yz	×	×	×
z->-z-z-z	×	×	○

(2) 終端記号列 (x)->y の非終端記号 L に対する構文木を書きなさい。そのような構文木が存在しない場合は「構文木なし」と書きなさい。(4 点)

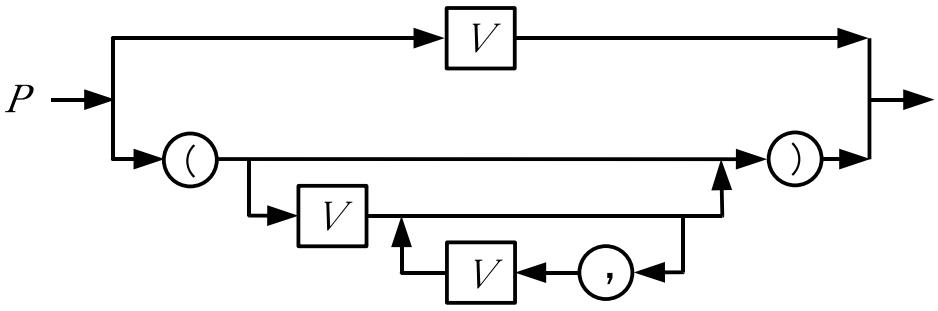


学籍番号　　氏名　　**(裏面に続く)**

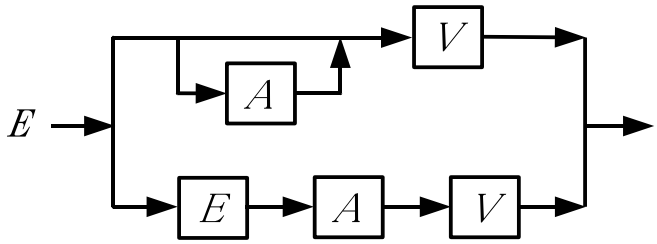
この定期試験の採点結果と科目の総合成績は、8 月 8 日 (土) までに Email でお知らせします。
このお知らせが不要な受講者は右の「不要」を丸で囲ってください。

不要

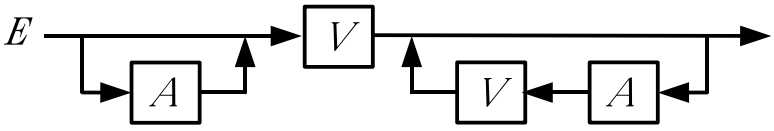
(3) 非終端記号 P の構文図を書きなさい。ただし、P の生成規則の右辺に現れている非終端記号の構文図を埋め込む必要はありません。(4 点)



(4) 非終端記号 E の構文図を書きなさい。ただし、E の生成規則の右辺に現れている非終端記号の構文図を埋め込む必要はありません。(4 点)



(5) 非終端記号 E の構文図を決定的な構文図に書き換えなさい。(4 点)



(6) 次ページの C プログラムは、標準入力 (キーボード) から、文字列と改行文字を入力すると、入力された文字列が非終端記号 L から導出できるかどうかを判定するプログラムの一部である。関数 P、E、L の定義を補って、このプログラムを完成しなさい。(26 点)

(次ページに続く)

```
#include <stdio.h>
#include <stdlib.h>

extern void error(void);
extern void L(void);

int t;

void gettoken(void)
{
    t = getchar();
}

int main()
{
    gettoken();
    L();
    if (t != '\n')
        error();
    printf("OK!\n");
    return 0;
}
```

```
void error(void)
{
    printf("Error!\n");
    exit(1);
}

void V(void)
{
    if (t == 'x' || t == 'y' || t == 'z')
        gettoken();
    else
        error();
}

void A(void)
{
    if (t == '+' || t == '-')
        gettoken();
    else
        error();
}
```

(問題 I (6) の解答欄)

```
void P(void)
{
    switch (t) {
    case 'x': case 'y': case 'z':
        V();
        break;
    case '(':
        gettoken();
        if (t == 'x' || t == 'y' || t == 'z') {
            V();
            while (t == ',') {
                gettoken();
                V();
            }
        }
        if (t != ')')
            error();
        gettoken();
        break;
    default:
        error();
    }

}
```

学籍番号　　氏名

(裏面に続く)

(問題 I (6) の解答欄の続き)

```
void E(void)
{

        if (t == '+' || t == '-')
            A();
        V();
        while (t == '+' || t == '-') {
            A();
            V();
        }

}

void L(void)
{

        P();
        if (t != '-')
            error();
        gettoken();
        if (t != '>')
            error();
        gettoken();
        E();

}
```

(次ページに問題 II)

II　次は Minimum C のソースプログラムと、それをコンパイルして得られた MVM の機械語プログラムである。

ソースプログラム

```
int b, n;

rev(n)
{
    int m;

    m = n / b;
    if (m >= 1) {
        int s;

        s = b;
        while ( m >= b ) {
            s = s * b;
            m = m / b;
        }
        n = n % b * s + rev(n / b);
    }
    return n;
}

main()
{
    n = 1;
    b = 10;
    while (n != 0) {
        input n;
        print rev(n);
    }
}
```

コンパイル結果

```
 0  ADD R13,R14,R0
 4  LDI R11,300
 8  CALL R0,220
12  EXIT R1
16  PUSH R13
20  ADD R13,R14,R0
24  SUBI R14,R14,4
28  LD R1,R13,8
32  LD R2,R11,0
36  DIV R1,R1,R2
40  ST R1,R13,-4
44  LD R1,R13,-4
48  LDI R2,1
52  SUB R0,R1,R2
56  JPLT R0,192
60  SUBI R14,R14,4
64  LD R1,R11,0
68  ST R1,R13,-8
72  LD R1,R13,-4
76  LD R2,R11,0
80  SUB R0,R1,R2
84  JPLT R0,124
88  LD R1,R13,-8
92  LD R2,R11,0
96  MUL R1,R1,R2
100 ST R1,R13,-8
104 LD R1,R13,-4
108 LD R2,R11,0
112 DIV R1,R1,R2
116 ST R1,R13,-4
120 JP R0,72
124 LD R1,R13,8
128 LD R2,R11,0
132 MOD R1,R1,R2
136 LD R2,R13,-8
140 MUL R1,R1,R2
144 PUSH R1
148 LD R1,R13,8
152 LD R2,R11,0
156 DIV R1,R1,R2
160 PUSH R1
164 CALL R0, 16
168 ADDI R14,R14,4
172 ADD R2,R1,R0
176 POP R1
180 ADD R1,R1,R2
184 ST R1,R13,8
188 ADDI R14,R14,4
192 LD R1,R13,8
196 ADD R14,R13,R0
200 POP R13
204 POP R15
208 ADDI R14,R14,4
212 POP R13
216 POP R15
220 PUSH R13
224 ADD R13,R14,R0
228 LDI R1,1
232 ST R1,R11,4
236 LDI R1,10
240 ST R1,R11,0
244 LD R1,R11,4
248 LDI R2,0
252 SUB R0,R1,R2
256 JPE R0,292
260 IN R1
264 ST R1,R11,4
268 LD R1,R11,4
272 PUSH R1
276 CALL R0,16
280 ADDI R14,R14,4
284 OUT R1
288 JP R0,244
292 POP R13
296 POP R15
```

学籍番号　　氏名

（裏面に続く）

(1) ソースプログラムとコンパイル結果が対応するように、前ページの空欄部分を補いなさい。(30 点)

(2) この機械語プログラムを起動して、まず、キーボードから 456 という入力を与えたとする。192 番地の機械語命令が初めて実行された直後の MVM のスタックポインタ (R14) の値は 1048504 であった。この時点でのスタック中のデータ (整数値) を下図の空欄に補いなさい。ただし、下図の 1 つの欄は、それぞれ 32bit (4byte) のデータを表すものとする。(12 点)

MVM のメモリ (スタック) の内容

アドレス	
	⋮
1048504	0
1048508	1048532
1048512	168
1048516	4
1048520	50
1048524	10
1048528	4
1048532	1048556
1048536	168
1048540	45
1048544	600
1048548	100
1048552	4
1048556	1048568
1048560	280
1048564	456
1048568	1048576
1048572	12
1048576	

(3) (2) の時点での R1、R11、R13 (フレームポインタ) の値を下の欄に書き込みなさい。(6 点)

R1	4
R11	300
R13	1048508

(定期試験問題終り)