

I 拡張 BNF 記法で書かれた次のような文法について以下の問いに答えなさい。

$$\langle B \rangle ::= "0" \mid "1" .$$

$$\langle C \rangle ::= \langle P \rangle \mid \langle M \rangle .$$

$$\langle L \rangle ::= "(" \langle B \rangle \{ " , " \langle B \rangle \} ")" .$$

$$\langle M \rangle ::= "[" \{ \langle B \rangle " , " \} \langle L \rangle \{ " , " \langle P \rangle \} "]" .$$

$$\langle P \rangle ::= \langle L \rangle \mid \langle B \rangle .$$

$$\langle E \rangle ::= [\langle E \rangle " , "] \langle C \rangle .$$

(1) 次の表中の終端記号列が、非終端記号 P 、 M 、 E 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。(12点)

終端記号列	P	M	E
0			
0,0,1			
(0),1,0			
(0,1,1)			
1,0,{0}			
(1,(0),1)			
{1,1,0}			
{1,(1),1}			

(2) 終端記号列 $\{(1,0)\}$ の非終端記号 C 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7日(月)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 M の構文図を書きなさい。(4点)

(4) 非終端記号 E の構文図を書きなさい。(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 E 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 C 、 M 、 E の定義をそれぞれ補って、このプログラムを完成しなさい。(24点)

(次ページに続く)

```

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

extern void B(void);
extern void L(void);
extern void P(void);
extern void C(void);
extern void M(void);
extern void E(void);

int t;

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

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

void B(void) {
    if (t == '0' || t == '1')
        gettoken();
    else
        error();
}

```

```

void L(void) {
    if (t != '(')
        error();
    gettoken();
    B();
    while (t == ',') {
        gettoken();
        B();
    }
    if (t != ')')
        error();
    gettoken();
}

void P(void) {
    if (t == '(')
        L();
    else
        B();
}

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

```

```

void C(void)
{

```

```

}

```

(問題 I (6) の解答欄の続き)

```
void M(void)
{
```

```
}
```

```
void E(void)
{
```

```
}
```

(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 n;

fib(n)
{
    if (  )
        return n;
    return fib(n-1) + fib(n-2);
}

main()
{
    input n;
    print sum(n);
}

sum(n)
{
    int s;

    s = 0;
    while (n > 0) {
        int t;

        t = fib();
        n = n - 1;
        s = s + t;
    }
    return ;
}

```

コンパイル結果

```

0  ADD R13,R14,R0
4  LDI R11, 
8  CALL R0, 
12 EXIT R1
16 PUSH R13
20 ADD R13,R14,R0
24 LD R1,R13,8
28 LDI R2,1
32 SUB R0,R1,R2
36 JPGT R0,52
40 LD R1,R13,8
44 POP R13
48 POP R15
52 LD R1,R13,8
56 LDI R2,1
60 SUB R1,R1,R2
64 PUSH R1
68 CALL R0, 
72 ADDI R14,R14,4
76 
80 LD R1,R13,8

```

```

84 LDI R2,2
88 SUB R1,R1,R2
92 PUSH R1
96 CALL R0,16
100 ADDI R14,R14,4
104 ADD R2,R1,R0
108 
112 ADD R1,R1,R2
116 POP R13
120 POP R15
124 POP R13
128 POP R15
132 PUSH R13
136 ADD R13,R14,R0
140 IN R1
144 ST R1,R11,0
148 LD R1,R11,0
152 
156 CALL R0,176
160 
164 OUT R1
168 POP R13
172 POP R15
176 PUSH R13
180 ADD R13,R14,R0
184 SUBI R14,R14,4
188 LDI R1,0
192 
196 LD R1,R13,8
200 LDI R2,0
204 SUB R0,R1,R2
208 
212 
216 LD R1,R13,8
220 PUSH R1
224 CALL R0,16
228 ADDI R14,R14,4
232 ST R1,R13,-8
236 LD R1,R13,8
240 LDI R2,1
244 SUB R1,R1,R2
248 ST R1,R13,8
252 LD R1,R13,-4
256 LD R2,R13,-8
260 ADD R1,R1,R2
264 ST R1,R13,-4
268 
272 JP R0, 
276 LD R1,R13,-4
280 ADD R14,R13,R0
284 POP R13
288 POP R15
292 ADDI R14,R14,4
296 POP R13
300 POP R15

```

(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、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、キーボードから、5 という数値を入力したとする。初めて 32 番地の機械語命令が実行された直後の MVM のスタックポインタ (R14) の値は 1048536 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	⋮
1048536	1048556
1048540	
1048544	
1048548	0
1048552	
1048556	
1048560	
1048564	
1048568	1048576
1048572	12

$$2^{20} = 1048576$$

- (3) (2) の時点での R1、R2、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	
R2	
R13	