

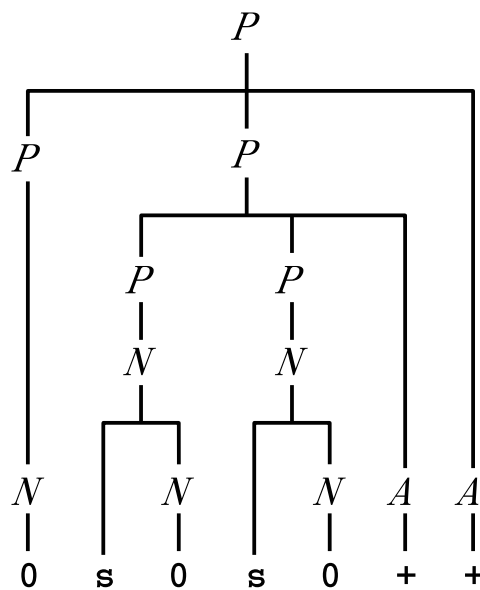
I 拡張 BNF 記法で書かれた次のような文法について以下の問いに答えなさい。

$$\begin{aligned}\langle N \rangle &::= \text{"0"} \mid \text{"s"} \langle N \rangle. \\ \langle A \rangle &::= \text{"*"} \mid \text{"+"}. \\ \langle P \rangle &::= \langle N \rangle \mid \langle P \rangle \langle P \rangle \langle A \rangle.\end{aligned}$$

(1) 次の表中の終端記号列が、非終端記号 N や P 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	N	P
0	○	○
s	×	×
00	×	×
s00	×	×
0s0	×	×
00*00*	×	×
ss0ss0+	×	○
s00+s00+*	×	○

(2) 終端記号列 0s0s0++ の非終端記号 P 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

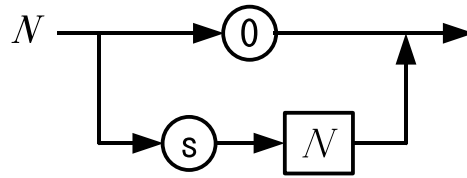
(裏面に続く)

この定期試験の採点結果と科目の総合成績は、8月8日(月)までに Email でお知らせします。

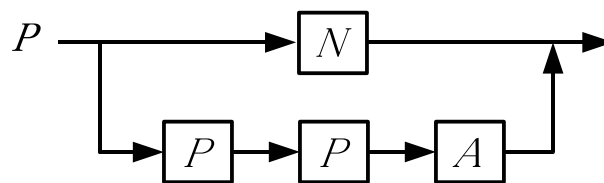
このお知らせが不要な受講者は右の「不要」を丸で囲ってください。

不要

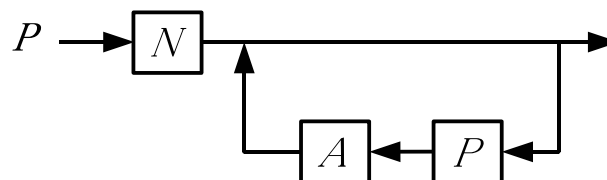
(3) 非終端記号 N の構文図を書きなさい。(4点)



(4) 非終端記号 P の構文図を書きなさい。(4点)



(5) 非終端記号 P 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 P 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 N 、 A 、 P 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)

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

extern void N(void);
extern void A(void);
extern void P(void);

int t;

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

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

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

```
void N(void)
{
```

```
    switch (t) {
    case '0':
        gettoken();
        break;
    case 's':
        gettoken();
        N();
        break;
    default:
        error();
    }
}
```

```
}
```

(問題 I (6) の解答欄の続き)

```
void A(void)
{
```

```
    switch (t) {
    case '*' : case '+' :
        gettoken();
        break;
    default:
        error();
    }
```

```
}
```

```
void P(void)
{
```

```
    N();
    while (t == '0' || t == 's') {
        P();
        A();
    }
```

```
}
```

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

```

gcd(x, y)
{
    while (  ) {
        int t;

        while (y >= x) {
            y = y - x;
        }
        t = x;
        x = y;
        y = t;
    }
    return y;
}

int p, q;

main()
{
    input p;
    input q;
    print p*q/gcd(p, q);
}

```

コンパイル結果

```

0  ADD R13,R14,R0
4  LDI R11,220
8  CALL R0, 
12 EXIT R1
16 PUSH R13
20 ADD R13,R14,R0
24 LD R1,R13,12
28 LDI R2,0
32 SUB R0,R1,R2
36 JPE R0,112
40 
44 LD R1,R13,8
48 LD R2,R13,12
52 SUB R0,R1,R2
56 

```

```

60 LD R1,R13,8
64 LD R2,R13,12
68 SUB R1,R1,R2
72 ST R1,R13,8
76 JP R0,44
80 
84 
88 LD R1,R13,8
92 ST R1,R13,12
96 LD R1,R13,-4
100 ST R1,R13,8
104 
108 JP R0, 
112 
116 POP R13
120 POP R15
124 POP R13
128 POP R15
132 PUSH R13
136 ADD R13,R14,R0
140 
144 
148 IN R1
152 ST R1,R11,4
156 LD R1,R11,0
160 LD R2,R11,4
164 MUL R1,R1,R2
168 
172 LD R1,R11,0
176 PUSH R1
180 LD R1,R11,4
184 PUSH R1
188 CALL R0,16
192 
196 ADD R2,R1,R0
200 
204 
208 OUT R1
212 POP R13
216 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、順に 14 と 20 という 2 つの数値を入力したとする。初めて 84 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 1048544 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	⋮
1048544	14
1048548	1048568
1048552	192
1048556	6
1048560	14
1048564	280
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	14
R11	220
R13	1048548