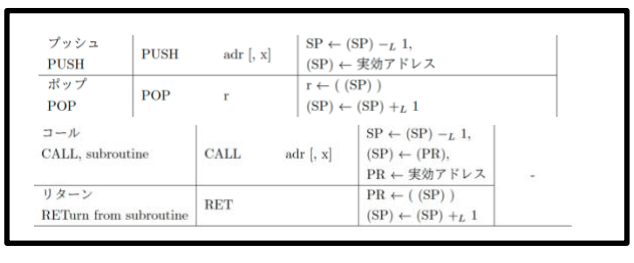
言語処理工学 A　期末テスト

2021 年 11 月 26 日　井上克郎

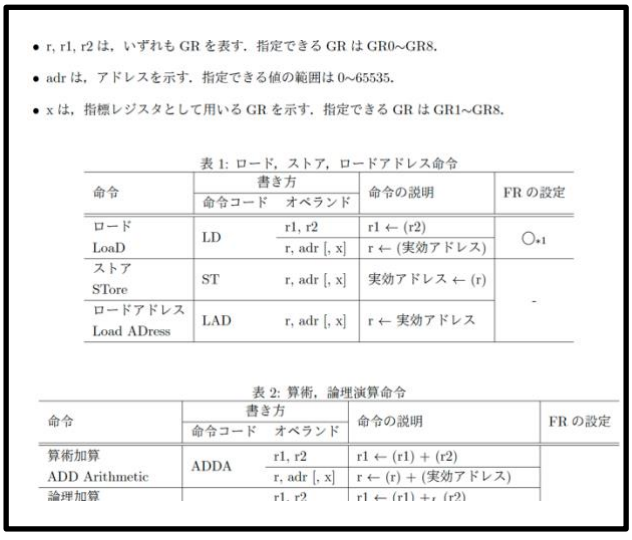
ノート教科書持ち込みなし。[1]は解答用紙表紙、[2]は折りたたんだ内側 2 枚に、[3] は裏表紙に解答を書くこと。間違った場所に書いた場合は**減点する**。

[1] 次のコードは、ある簡単な PASCAL プログラムを CASL にコンパイルしたものである。元の PASCAL プログラムを書け。ユーザー定義識別子名（プログラム、関数、変数名など）は適当に決めて良い。また、このプログラムに 10 を入力した際の実行結果（整数 1 つ）を書け。(20 点)

```
CASL    START   BEGIN
BEGIN   LAD     GR6, 0  ; reset write buffer
        LAD     GR7, LIBBUF     ; set output buffer address
        LAD     GR5, 0  ; set FP(GR5) null
        JUMP    LLMAIN  ; go to the main program
; beginning of subprogram
LLf     NOP     ; entry of subprogram
        LD      GR1,1,GR5       ; get actual parameter
        PUSH    0,GR1   ; Save 1st operand in GR1 for expression comparison
        LAD     GR1,0   ; set integer constant as 2nd operand
        POP     GR2     ; Restore the 1st operant to GR1 for comparison
        CPA     GR2,GR1 ;Execute comparison and set flags
        LAD     GR1,1   ; Initialize GR1 with True, no flag change
        JZE     LB1     ; jump EQUAL
        XOR     GR1,GR1 ; set GR1 to False (zero)
LB1     NOP
                ; End of Exp comparison
        AND     GR1,GR1 ; check if's condition
        JZE     LB2     ; jump if not true
        LAD     GR1,0   ; set integer constant
        ST      GR1,ZZresult    ; complete assignment
        JUMP    LB3     ; exit of if statement
LB2     NOP     ;
        LD      GR1,1,GR5       ; get local variable
        PUSH    0,GR1   ; start -
        LAD     GR1,1   ; set integer constant
        LD      GR2,GR1
        POP     GR1
        SUBA    GR1,GR2 ; End of -
        PUSH    0,GR1   ; set actual parameter to stack top
        PUSH    0,GR5   ; save FP (GR5) to stack
        LD      GR5,GR8 ; set new FP (GR5)
        CALL    LLf
        POP     GR5     ; restore FP(GR5)
        LAD     GR8,1,GR8       ; discard parameters
        LD      GR1,ZZresult    ; load global variable
        PUSH    0,GR1   ; start +
        LD      GR1,1,GR5       ; get local variable
        LD      GR2,GR1
        POP     GR1
        ADDA    GR1,GR2 ; End of +
        ST      GR1,ZZresult    ; complete assignment
LB3     NOP     ;
        RET
; end of subprogram
LLMAIN NOP      ; entry of main program
        LAD     GR2,ZZi ; set global variable address to GR2
        CALL    RDINT   ; read one integer to (GR2)
        LD      GR1,ZZi ; load global variable
        PUSH    0,GR1   ; set actual parameter to stack top
        PUSH    0,GR5   ; save FP (GR5) to stack
        LD      GR5,GR8 ; set new FP (GR5)
        CALL    LLf
        POP     GR5     ; restore FP(GR5)
        LAD     GR8,1,GR8       ; discard parameters
        LD      GR1,ZZresult    ; load global variable
        LD      GR2,GR1
        CALL    WRTINT          ; write an integer
        CALL    WRTLN           ; write new line
        RET
; start of global variable area
ZZi     DS      1
ZZresult DS     1
; end of global variable area
LIBBUF  DS      256
        END
 ...
```



プッシュ PUSH	PUSH	adr [, x]	SP ← (SP) −L 1, (SP) ← 実効アドレス
ポップ POP	POP	r	r ← ((SP)) (SP) ← (SP) +L 1
コール CALL, subroutine	CALL	adr [, x]	SP ← (SP) −L 1, (SP) ← (PR), PR ← 実効アドレス
リターン RETurn from subroutine	RET		PR ← ((SP)) (SP) ← (SP) +L 1



- r, r1, r2 は，いずれも GR を表す．指定できる GR は GR0∼GR8.
- adr は，アドレスを示す．指定できる値の範囲は 0∼65535.
- x は，指標レジスタとして用いる GR を示す．指定できる GR は GR1∼GR8.

表 1: ロード，ストア，ロードアドレス命令

命令	書き方 命令コード	オペランド	命令の説明	FR の設定
ロード LoaD	LD	r1, r2	r1 ← (r2)	○*1
		r, adr [, x]	r ← (実効アドレス)	
ストア STore	ST	r, adr [, x]	実効アドレス ← (r)	-
ロードアドレス Load ADress	LAD	r, adr [, x]	r ← 実効アドレス	

表 2: 算術，論理演算命令

命令	書き方 命令コード	オペランド	命令の説明	FR の設定
算術加算 ADD Arithmetic	ADDA	r1, r2	r1 ← (r1) + (r2)	
		r, adr [, x]	r ← (r) + (実効アドレス)	
論理加算		r1, r2	r1 ← (r1) +L (r2)	

［2］次の 3 番地コードに関して答えよ。（解答用紙２－３ページに答えを書くこと）
（各 10 点）

```
①      x = 0
②      y = 1
③ L1: if x>10 goto L3
④      i = 1
⑤      x = x + i
⑥      if y=1 goto L2
⑦      x = 0
⑧ L2: goto L1
⑨ L3: y = x + 1
```

(1) このプログラムの基本ブロックに分け、プログラムの上から順に、各ブロックに番号 B1～をつけ、各ブロックに入るコードを行番号で明示せよ。
(2) 各基本ブロックを頂点としたフローグラフを書け。
(3) 得られたフローグラフの支配木（dominator tree）を書け。
(4) フローグラフ中の各バックエッジを挙げ、それぞれが構成するループの各頂点を示せ。
(5) 各基本ブロックの Gen 集合、Kill 集合を求めよ。
(6) 各基本ブロックの入り口（IN)、出口（OUT）で出現しうる変数定義の行番号の集合（データフロー方程式の解）を示せ。（途中結果も示せ）

［3］上記［2］のプログラムにおいて、最適化が適用できる２箇所を探し、それぞれについて、どのような書き換えが可能か、また、なぜそれが安全にできるか、説明せよ。［2］の結果を使っても使わなくてもよい。（4 ページめに書くこと）（10 点×２）

解答例

[１]
以下のようなプログラム。

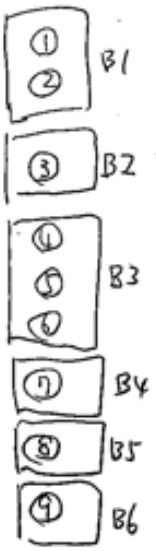
```
program cal(input,output);
var i,result：  integer;

procedure f(n      : integer);

begin
    if (n=0) then result := 0
    else begin
        f(n-1);
        result:= result+n
        end
end;

begin
    readln(i);
    f(i);
    writeln(result)

end.
```

実行結果
55

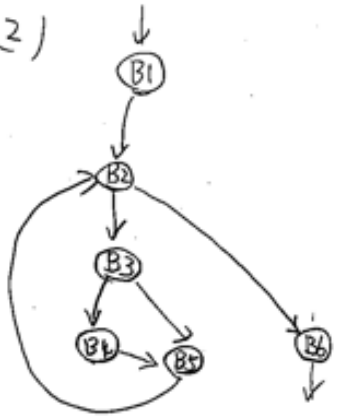
［２］下に添付

［３］例えば以下のような解答
＊ ⑥に到達する生きてる y は②のみ、よって y =1 が常に成り立つので⑥の if 文を goto L2 という無条件分岐文に置き換え可能
＊ 上記最適化により⑦は実行されない命令となり削除可能
＊ ⑤に到達する i は④の定義のみ、よって x+1 と書き換え可能、更にこの文は inc x に書き換え可能
＊ 上記以外の文で i の参照は無いので、④は不要
など

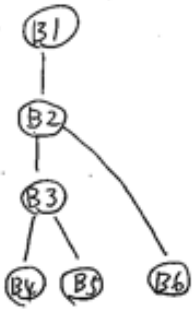
(1)



(2)



(3)



(4) (B5, B2) : {B2, B3, B4, B5}

(5)

BLK#	Gen	Kill
1	1, 2	5, 7, 9
2	—	—
3	4, 5	1, 7
4	7	1, 5
5	—	—
6	9	2

(6) initial (step 0)

BLK#	IN	OUT
1	—	1, 2
2	—	—
3	—	4, 5
4	—	7
5	—	—
6	—	9

step 1

BLK#	IN	OUT
1	—	1, 2
2	1, 2	1, 2
3	—	4, 5
4	4, 5	4, 7
5	4, 5, 7	4, 5, 7
6	—	9

step 2

BLK#	IN	OUT
1	—	1, 2
2	1, 2, 4, 5, 7	1, 2, 4, 5, 7
3	1, 2	2, 4, 5
4	4, 5	4, 7
5	4, 5, 7	4, 5, 7
6	1, 2	1, 9

step 3

BLK#	IN	OUT
1	—	1, 2
2	1, 2, 4, 5, 7	1, 2, 4, 5, 7
3	1, 2, 4, 5, 7	2, 4, 5
4	2, 4, 5	2, 4, 7
5	2, 4, 5, 7	2, 4, 5, 7
6	1, 2, 4, 5, 7	1, 4, 5, 7, 9

step 4

BLK#	IN	OUT
1	—	1, 2
2	1, 2, 4, 5, 7	1, 2, 4, 5, 7
3	1, 2, 4, 5, 7	2, 4, 5
4	2, 4, 5	2, 4, 7
5	2, 4, 5, 7	2, 4, 5, 7
6	1, 2, 4, 5, 7	1, 4, 5, 7, 9

同じ, よって停止