

プログラミング言語論 第1回小テスト 解答例

問1

Little Quilt 言語の以下の各式 (1), (2), (3) が表すキルトを図示せよ。a, b, turn, sew の意味は以下の通りである。その他の Little Quilt 言語の構文要素 (let 式、val 宣言、fun 宣言) の意味は講義で説明したものとする。

- a, b は図??, 図??のキルトを表す。

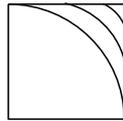


図 1: a が表すキルト

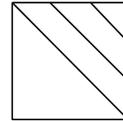
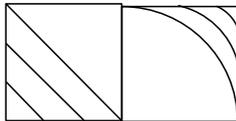


図 2: b が表すキルト

- turn (e) : キルト e を 90 度右回転させたキルトを表す。
- sew (e₁, e₂) : 高さが同じキルト e₁, e₂ を左右に並べ、縫い合わせたキルトを表す (左が e₁、右が e₂)。

(1) sew (turn (turn (b)), a)



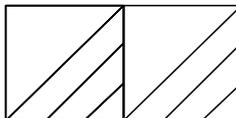
(2) let

```
    val x = turn (b)
```

```
in
```

```
    sew (x,x)
```

```
end
```



(3) let

```
    fun unturn (x) = turn (turn (turn (x)))
```

```
    fun pile (x,y) = unturn (sew (turn (y), turn (x)))
```

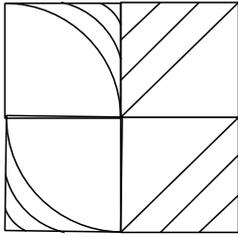
```
    val aa = pile (a, turn (turn (a)))
```

```
    val bb = pile (unturn (b), turn (b))
```

```
in
```

```
    sew (aa, bb)
```

```
end
```



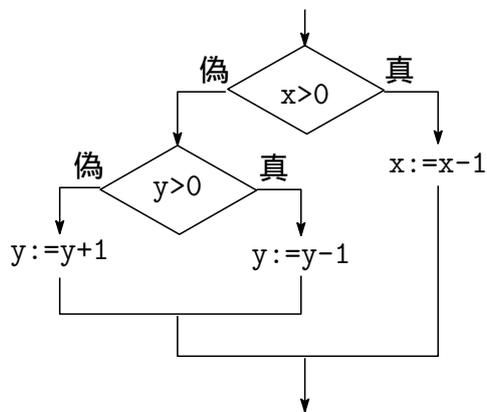
問2 命令型言語の制御フローについて以下の各問に答えよ。

(1) 以下のプログラム断片の制御フローを図示せよ。

```

if x>0 then x := x - 1
else if y>0 then y := y - 1
else y := y + 1

```

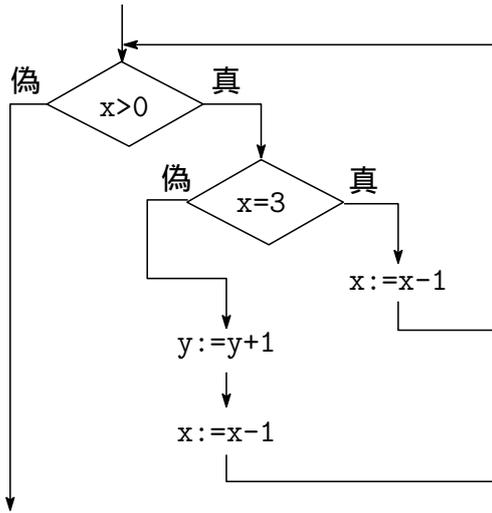


(2) 以下のプログラム断片の制御フローを図示せよ。

```

while x>0 do
begin
if x=3 then
begin
x := x - 1;
continue
end;
y := y + 1;
x := x - 1
end

```



問3 以下の Hoare triple(1), (2) を講義中に提示した規則を使って導け。

$$(1) \{a = 3\} a := a + 1 \{a = 4\}$$

$$\frac{a = 3 \Rightarrow a + 1 = 4 \quad \frac{}{\{a + 1 = 4\} a := a + 1 \{a = 4\}} \text{(assign)} \quad a = 4 \Rightarrow a = 4}{\{a = 3\} a := a + 1 \{a = 4\}} \text{(conseq)}$$

授業時にも言った通り、上記証明の中で、 $a = 4 \Rightarrow a = 4$ の部分は \Rightarrow の左右が同じなので省略し、以下のように書いてもよい。

$$\frac{a = 3 \Rightarrow a + 1 = 4 \quad \frac{}{\{a + 1 = 4\} a := a + 1 \{a = 4\}} \text{(assign)}}{\{a = 3\} a := a + 1 \{a = 4\}} \text{(conseq)}$$

$$(2) \{a = 3\} a := a + 1; a := a + 2 \{a = 6\}$$

$$\frac{\frac{a = 3 \Rightarrow a + 1 = 4 \quad \frac{}{\{a + 1 = 4\} a := a + 1 \{a = 4\}} \text{(assign)}}{\{a = 3\} a := a + 1 \{a = 4\}} \text{(conseq)} \quad \frac{a = 4 \Rightarrow a + 2 = 6 \quad \frac{}{\{a + 2 = 6\} a := a + 2 \{a = 6\}} \text{(assign)}}{\{a = 4\} a := a + 2 \{a = 6\}} \text{(conseq)}}{\{a = 3\} a := a + 1; a := a + 2 \{a = 6\}} \text{(composition)}$$

この証明では、2カ所の consequence rule の適用のところ、 $a = 4 \Rightarrow a = 4$ と $a = 6 \Rightarrow a = 6$ を省略した。

$$(3) \{x = 5\} \text{ while } x > 0 \text{ do } x := x - 1 \{x = 0\}$$

スペースの関係で、2か所に分けて記述する。

$$\frac{\text{この部分は下に記述。} \quad \frac{x = 5 \Rightarrow x \geq 0 \quad \frac{}{\{x \geq 0\} \text{ while } x > 0 \text{ do } x := x - 1 \{x \geq 0 \wedge \neg x > 0\}} \text{(conseq)} \quad x \geq 0 \wedge \neg x > 0 \Rightarrow x = 0}{\{x = 5\} \text{ while } x > 0 \text{ do } x := x - 1 \{x = 0\}} \text{(conseq)}$$

$$\frac{x \geq 0 \wedge x > 0 \Rightarrow x - 1 \geq 0 \quad \frac{}{\{x - 1 \geq 0\} x := x - 1 \{x \geq 0\}} \text{(assign)} \quad x \geq 0 \Rightarrow x \geq 0}{\frac{}{\{x \geq 0 \wedge x > 0\} x := x - 1 \{x \geq 0\}} \text{(conseq)} \quad \frac{}{\{x \geq 0\} \text{ while } x > 0 \text{ do } x := x - 1 \{x \geq 0 \wedge \neg x > 0\}} \text{(while)}}$$

この下側の証明図でも、 $x \geq 0 \Rightarrow x \geq 0$ の部分は \Rightarrow の左右が同じなので、省略して以下のように書いてよい。

$$\frac{\frac{x \geq 0 \wedge x > 0 \Rightarrow x - 1 \geq 0 \quad \overline{\{x - 1 \geq 0\} x := x - 1 \{x \geq 0\}} \text{ (assign)}}{\{x \geq 0 \wedge x > 0\} x := x - 1 \{x \geq 0\}} \text{ (conseq)}}{\{x \geq 0\} \text{ while } x > 0 \text{ do } x := x - 1 \{x \geq 0 \wedge \neg x > 0\}} \text{ (while)}$$

上記証明において、assignment axiom を assign, consequence rule を conseq, while rule を while, composition rule を composition と略記した。

問4 以下の Pascal プログラムを実行したときの出力結果を示せ。手続きの仮引数に var が付いている場合、call by reference であることを表す。writeln は引数の値を出力後改行する。

```

program test;
var x : integer;
var y : integer;
procedure swap
  (var x: integer;
   var y : integer);
var z : integer;
begin
  z := x; x := y; y := z
end;
begin
  x := 3;
  y := 4;
  swap (x,y);
  writeln (x);
  writeln (y)
end.

```

4
3

問5 以下の Pascal プログラムを実行したときの出力結果を示せ。Pascal における変数の有効範囲 (scope) の定め方は static scope であることに注意せよ。

```

program P;
var n : char;
procedure W;
begin
  writeln(n)
end;
procedure D;
var n : char;
begin
  n := 'D';
  W
end;
begin
  n := 'L';
  W;
  D
end.

```

L
L

問6 以下の (1), (2) のプログラムの意味 (状態の変化) を、講義中に提示した規則にしたがって示せ。ただし、これらのプログラムは、講義中で意味の定義を紹介するとき定義した、C の非常に小さなサブセットによるプログラムである。(1), (2) のプログラムの実行前の状態は、いずれも $\sigma = \{(X, 3), (Y, 1), (Z, 0)\}$ とする。

(1) $Z=(X+4);$

$$\frac{\frac{\langle X, \sigma \rangle \rightarrow 3 \quad \langle 4, \sigma \rangle \rightarrow 4}{\langle (X+4), \sigma \rangle \rightarrow 7}}{\langle Z = (X+4);, \sigma \rangle \rightarrow \sigma[7/Z]}$$

上記より、状態 σ においてプログラム $Z=(X+4);$ を実行すると、状態は

$$\sigma[7/Z] = \{(X, 3), (Y, 1), (Z, 7)\}$$

になる。

(2) $\text{while}(Y)\{Y=(Y-1);\}$

$$\frac{\frac{\frac{\langle Y, \sigma \rangle \rightarrow 1 \quad \langle 1, \sigma \rangle \rightarrow 1}{\langle (Y-1), \sigma \rangle \rightarrow 0} \quad \langle Y, \sigma[0/Y] \rangle \rightarrow 0}{\langle Y, \sigma \rangle \rightarrow 1 \quad \langle Y = (Y-1);, \sigma \rangle \rightarrow \sigma[0/Y]} \quad \langle \text{while}(Y)\{Y = (Y-1);\}, \sigma[0/Y] \rangle \rightarrow \sigma[0/Y]}{\langle \text{while}(Y)\{Y = (Y-1);\}, \sigma \rangle \rightarrow \sigma[0/Y]}$$

上記より、状態 σ においてプログラム $\text{while}(Y)\{Y=(Y-1);\}$ を実行すると、状態は

$$\sigma[0/Y] = \{(X, 3), (Y, 0), (Z, 0)\}$$

になる。