1 【必須問題】アルゴリズムとプログラミング　（情報工学 1/16）

配点：(1) 26 点，(2-1) 12 点，(2-2) 36 点，(2-3) 26 点，(3) 25 点

図 1 は ANSI-C 準拠である C 言語のプログラム（program）である．このプログラムは，34 行目で宣言（declare）される struct item_t 型の配列（array）data 内の要素（element）を，関数（function）sort を用いて int 型である key の値に従って整列（sort）したのち，整列した結果（result）を関数 show で出力（output）する．以下の各問に答えよ．

(1) 関数 show は，引数（argument）である struct item_t 型の配列 data の要素のうち，int 型である key と char 型である val の値（value）を先頭から順に 28 行目の関数 printf で出力する．そのような動作となるよう，プログラム中の空欄（ア）および（イ）に適切な式（expression）を埋めよ．

(2) 関数 sort は，引数である配列 data を入力データ（input data）として整列を実行する．以下の小問に答えよ．

(2-1) 関数 sort の引数 data の内容（34 行目の（ウ））が

```
{5, 'a'}, {2, 'd'}, {6, 'e'}, {4, 'h'}, {1, 'b'}, {8, 'f'}, {3, 'c'}, {7, 'g'}
```

であったとする．この時，プログラムの標準出力（standard output）から得られる整列結果を答えよ．

(2-2) 関数 sort のうち，16 行目から 19 行目までの一連の代入は，配列の二つの要素の値を入れ替え操作（swap operation）によって変更する処理である．この一連の代入が一回起きることを，入れ替え操作が一回起きたとする．34 行目の（ウ）が以下の（a）～（d）であったとき，入れ替え操作の回数は（a）～（d）それぞれ何回となるか答えよ．

(a) `{1, 'a'}, {2, 'd'}, {4, 'e'}, {6, 'h'}, {8, 'b'}, {3, 'f'}, {5, 'c'}, {7, 'g'}`
(b) `{8, 'c'}, {2, 'f'}, {3, 'h'}, {4, 'g'}, {5, 'd'}, {6, 'e'}, {7, 'b'}, {1, 'c'}`
(c) `{8, 'c'}, {1, 'f'}, {2, 'h'}, {3, 'g'}, {4, 'd'}, {5, 'e'}, {6, 'b'}, {7, 'c'}`
(d) `{2, 'x'}, {4, 'y'}, {6, 'z'}, {8, 'w'}, {2, 's'}, {4, 't'}, {6, 'u'}, {8, 'v'}`

(2-3) 関数 sort におけるデータの整列は，入力データに依存して入れ替え操作の回数が大きく変化する．整列が終了するまでに，入れ替え操作が k 回（$0 \leq k \leq (MAX \times (MAX-1))/2$）起きるような入力データの条件は次の数式（formula）で与えられる．

$$|\{ <i,j> \mid 0 \leq i < j \leq MAX-1, \boxed{\text{(エ)}} \}| = \boxed{\text{(オ)}}$$

ただし，$|S|$ は集合（set）S の要素数，$<i,j>$ は順序対（ordered pair）を表す．なお，入力データの key の値を先頭の要素から順に$a_0, a_1, \ldots, a_{MAX-1}$とする．空欄（エ）および（オ）に適切な数式を埋めよ．

(3) key の値が同じ要素に対して，整列前の要素の並び順の前後関係が整列後も維持されるとき，安定な（stable）整列アルゴリズムという．関数 sort が実現しているアルゴリズムが安定であるかどうか，プログラムの箇所を指摘しつつ，理由も含めて説明せよ．

```
 1  #include <stdio.h>
 2
 3  #define MAX 8
 4
 5  struct item_t {
 6      int key; char val;
 7  };
 8
 9  void sort(struct item_t data[])
10  {
11      int i, j, tmp_key;
12      char tmp_val;
13
14      for (i = 1; i < MAX; i++)
15          for (j = i; j > 0 && data[j-1].key >= data[j].key; j--) {
16              tmp_key = data[j].key; tmp_val = data[j].val;
17              data[j].key = data[j-1].key;
18              data[j].val = data[j-1].val;
19              data[j-1].key = tmp_key; data[j-1].val = tmp_val;
20          }
21  }
22
23  void show(struct item_t data[])
24  {
25      int i = MAX;
26
27      while ( [(ア)] )
28          printf("(%d,%c) ", [(イ)].key, [(イ)].val);
29      printf("\n");
30  }
31
32  int main(void)
33  {
34      struct item_t data[MAX] = {[        (ウ)        ]};
35
36      sort(data);
37
38      show(data);
39
40      return 0;
41  }
```

図 1　プログラム

2 【必須問題】計算機システムとシステムプログラム （情報工学 3/16）

配点: (1-1-1) 7 点, (1-1-2) 7 点, (1-1-3) 7 点, (1-2) 20 点, (1-3) 20 点,
(2-1) 14 点, (2-2-1) 16 点, (2-2-2) 14 点, (2-2-3) 20 点

(1) 計算機のキャッシュメモリ (cache memory) に関する以下の各小問に答えよ.

(1-1) セット数 2 のセット連想マッピング方式 (2 ウェイ群連想マッピング方式; 2-way set associative mapping) のキャッシュメモリについて考える. キャッシュサイズ (cache size) は 16 [バイト], ブロックサイズ (block size) は 4 [バイト], アドレス (address) はバイト単位, アドレス長は 8 [ビット], ブロック置き換え方法は LRU (Least Recently Used), キャッシュメモリの初期状態は空とする. 以下のアドレス (2 進数表記) に対するメモリアクセス (memory access) が上から順番に発生した場合を考える.

01011011
01001010
01001001
00010101
01101001
00010100
01101001
01001010
01011010
00010101

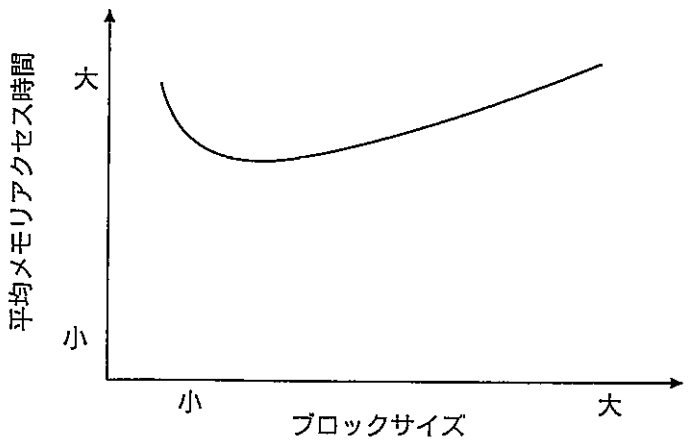
(1-1-1) 初期参照 (ブロックへの最初のアクセス) によるキャッシュミス (cache miss) の回数を求めよ.

(1-1-2) ブロックの置き換えを伴うキャッシュミスの回数を求めよ.

(1-1-3) キャッシュミス率を求めよ.

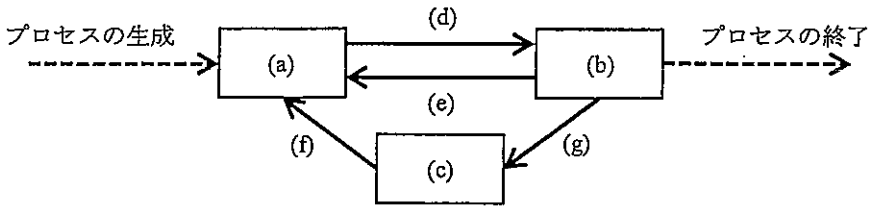
(1-2) キャッシュメモリはプログラムの局所性 (locality) を利用している. プログラムの局所性を二つ挙げよ. また, それぞれがキャッシュの効果にどのように寄与するかを説明せよ.

(1-3) 一般に, ブロックサイズと平均メモリアクセス時間 (mean memory access time) との関係は下図のような傾向を示す. ブロックサイズが小さい時と, ブロックサイズが大きい時に, 平均メモリアクセス時間が大きな値となる理由をそれぞれ説明せよ.



(2) 単一プロセッサ(uniprocessor)をもつマルチプログラミングシステム(multiprogramming system)に関する以下の各小問に答えよ.

(2-1) 単一プロセッサを持つマルチプログラミングシステムにおいて，プロセス(process)は，実行(running)状態，実行可能(ready)状態，待ち(wait または blocked)状態の三つの状態を持ち，プロセスが生成されてから終了するまでその 3 状態間を遷移する．次に示すプロセスの状態遷移図(state transition diagram)の(a)～(g)にあてはまるもっともふさわしい語句を下の選択肢から一つ選び，記号で答えよ.



選択肢

（ア）システムコール(system call) （イ）入出力処理の完了 （ウ）セグメンテーション(segmentation)
（エ）リモートプロシジャコール(remote procedure call) （オ）待ち状態 （カ）ページング(paging)
（キ）ディスパッチ(dispatch) （ク）実行状態 （ケ）コンパイル(compile) （コ）横取り(preemption)
（サ）実行可能状態 （シ）バッファリング(buffering) （ス）入出力処理の発生

(2-2) 表 1 にプロセス P1, P2, P3, P4 が生成される時刻，およびそれぞれのプロセスの処理時間を示す．下記の仮定に留意し，次の(2-2-1)～(2-2-3)に答えよ.

仮定

1. ある時刻に生成あるいは横取りされたプロセスは，その時刻に直ちに実行可能キュー(ready queue)に格納される．生成されたプロセスと横取りされたプロセスが，実行可能キューに同時刻に格納される場合には，生成されたプロセスが先に実行可能キューに格納される.
2. あるプロセスが終了あるいは横取りされた時刻から次のプロセスがプロセッサにディスパッチされる時刻までに要する処理時間（プロセス切り替えに関する処理時間）を 0 とする.
3. ラウンドロビン(round robin)方式において，タイムスライス(time slice)中にプロセスが終了したときは，直ちに次のプロセスがディスパッチされる.

(2-2-1) プロセスのスケジューリングアルゴリズム(scheduling algorithm)として，次の二つの方式，到着順(first come first service (served))方式，処理時間順(shortest processing time first)方式を用いた場合のプロセス P1, P2, P3, P4 のターンアラウンド時間(turnaround time)を求めよ.

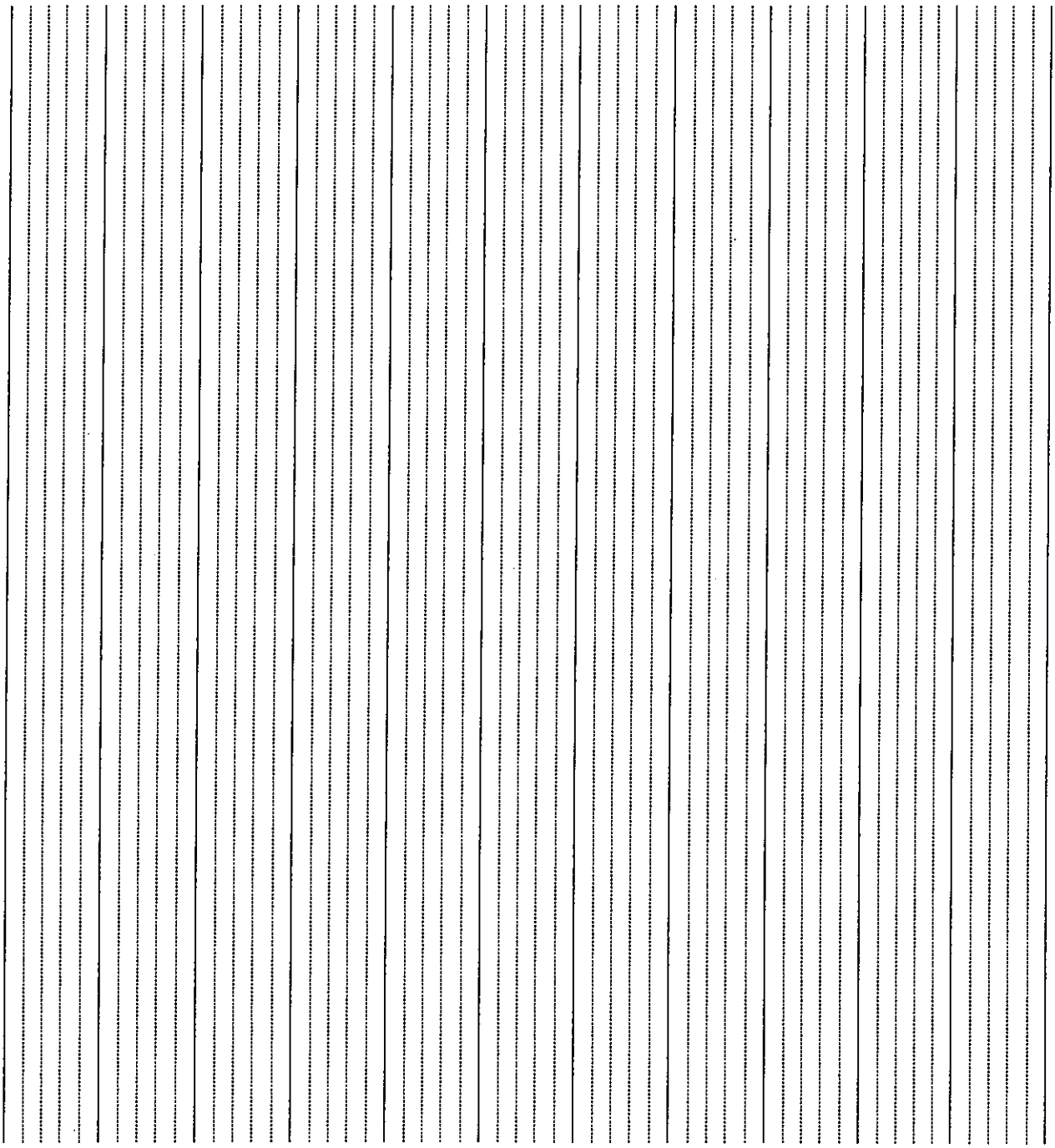
(2-2-2) プロセスのスケジューリングアルゴリズムとして，ラウンドロビン方式を用いる．(a) タイムスライスが 4 のときのプロセス P1, P2, P3, P4 の平均ターンアラウンド時間, (b) タイムスライスが 8 のときのプロセス P1, P2, P3, P4 の平均ターンアラウンド時間を求めよ.

(2-2-3) ラウンドロビン方式において，上記の仮定 2. を変更し，プロセス切り替えに関する処理時間を 0 より大きくしたとする．この場合，タイムスライスの値の大小によって平均ターンアラウンド時間がどう変化するか説明せよ.

表 1 プロセスの生成時刻と処理時間

プロセス	生成時刻	処理時間
P1	0	20
P2	8	40
P3	18	10
P4	24	30

（計算用紙）



3 【選択問題】離散構造

配点:(1) 28 点 (2-1) 17 点 (2-2) 20 点 (3) 30 点 (4-1) 15 点 (4-2) 15 点

論理式 (logic formula) の記号 $\Leftrightarrow$, $\Rightarrow$, $\wedge$, $\vee$, $\neg$ は，それぞれ，等価 (equivalence)，含意 (implication)，連言 (論理積)(conjunction, and)，選言 (論理和)(disjunction, or)，否定 (negation, not) を表す論理演算子とする．また，真 (true)，偽 (false) を表す，true, false の二値からなる集合を B とする．

(1) 一階述語論理 (first-order logic) における式 (formula) の解釈 (interpretation)I は (D, C, F, P) の 4 項組で与えられる．ここで，D は値集合，C は各定数記号への D の要素の割り当て，F は各 n 引数関数記号への $f : \mathsf{D}^n \to \mathsf{D}$ なる関数 f の割り当て，P は各 n 引数述語記号への $p : \mathsf{D}^n \to \mathsf{B}$ なる述語 p の割り当てである．

例えば一階述語論理式 $\forall x\ p(f(b,x),a)$ に対して，解釈 I_0 として
- D を非負整数 (nonnegative integer) 全体からなる集合とし，
- C として a, b それぞれへ非負整数値 0,1 を割り当て，
- F として 2 引数関数記号 $f(u,w)$ へ非負整数上の加算 $u+w$ を割り当て，
- P として 2 引数述語記号 $p(u,w)$ へ非負整数上の比較演算 $u > w$ を割り当てたとき (例えば $4 > 3$ の値は true である)，

式 $\forall x\ p(f(b,x),a)$ の解釈 I_0 のもとでの評価値は true となる．

以下の各式が恒真 (valid) か，恒真ではないが充足可能 (satisfiable) か，充足不能 (unsatisfiable) かを答えよ．恒真ではないが充足可能の場合は，値集合 D を $\{a, b\}$ で固定して，真にする解釈と偽にする解釈を 1 つずつ与えよ．

(a) $\forall x\ p(x) \Rightarrow \forall y\ p(y)$

(b) $\exists x\ p(x) \Rightarrow \forall x\ p(x)$

(c) $\neg\ \exists y\ p(y) \Leftrightarrow \forall x\ p(x)$

(d) $\forall x\ \exists y\ q(x,y) \Rightarrow \exists y\ \forall x\ q(x,y)$

(2) 以下で与えられる論理式 E が恒真であることを，まず，E の否定のスコーレム化 (Skolemization)，すなわち限量子 (quantifier) $\exists$ に関わる変数の消去，を行い，ついで，導出原理 (resolution principle) により，充足不能であることの確認を行うことによって示したい．

$A = p(f(g(f(g(g(a))))))$
$B = \forall x\ (p(f(g(x))) \Rightarrow p(x))$
$C = \forall x\ (p(g(f(x))) \Rightarrow p(x))$
$D = \exists x\ p(g(x))$
$E = (A \wedge B \wedge C) \Rightarrow D$
ただし，a は定数記号，f, g は関数記号，p は述語記号である．

以下の各小問に答えよ．

(2-1) $\neg E$ のスコーレム連言標準形 (Skolem conjunction normal form)E' を求めよ．導出過程は不要．E' 中では記号 A, B, C, D を用いないこと．

(2-2) 小問 (2-1) で得た論理式 E' をもとに，導出原理を用いて E' が充足不能であることを示せ．

(3) $X \cap Y = \emptyset$, $X \neq \emptyset$, $Y \neq \emptyset$ である任意の有限集合 X, Y に対し, $X \cup Y$ 上の以下の関係 R が, 反射律 (reflexive law), 反対称律 (antisymmetric law), 推移律 (transitive law) を満たすかどうかを述べ, その証明も与えよ. また R は順序関係 (order relation) かそうでないかを述べよ.

$$R = X \times Y \cup \{ (x, x) \mid x \in X \cup Y \}$$

(4) $\Sigma = \{\mathrm{a}, \mathrm{b}\}$ とする. Σ 上の長さ n $(n \geq 1)$ の文字列のうち, bb を部分文字列として含まない文字列の集合を L_n で表し, L_n の要素数を $|L_n|$ で表す. 以下の各小問に答えよ.

(4-1) 最後尾が w $(w \in \Sigma)$ である文字列からなる L_n の最大部分集合を $L_n(w)$ で表す. $n > 2$ のとき, $|L_n(\mathrm{a})|$, $|L_n(\mathrm{b})|$ それぞれを $|L_{n-1}|$, $|L_{n-2}|$ を用いて表せ.

(4-2) $|L_n|$ を n についての漸化式で表せ.

配点: (1) 15 点, (2) 30 点, (3) 20 点, (4-1) 35 点, (4-2) 25 点

(1) 以下に示す各言語 $L_1, ..., L_6$ が文脈自由言語 (context-free language) か否かを，○ (文脈自由言語である) か× (文脈自由言語でない) で答えよ．証明は不要である．

- $L_1 = \{ww \mid w \in \{a,b\}^*\}$
- $L_2 = \{ww^R \mid w \in \{a,b\}^*,\ w^R$は w の反転 (reversal)$\}$
- $L_3 = \{a^n b^m c^n d^m \mid n \geq 1$ かつ $m \geq 1\}$
- $L_4 = \{a^n b^m c^m d^n \mid n \geq 1$ かつ $m \geq 1\}$
- $L_5 = \{a^n b^n c^m d^m \mid n \geq 1$ かつ $m \geq 1\}$
- $L_6 = \{a^n b^n c^n \mid n \geq 1\}$

(2) 文脈自由文法 (context-free grammar) G_1 を以下の通りとする．この文法には単記号規則 (unit production), すなわち非終端記号 (non-terminal symbols) A と B に対して $A \to B$ の形の生成規則 (generating rule) が含まれている．このとき新たな非終端記号を追加せずに G_1 から単記号規則を除去した等価 (equivalent) な文法における，非終端記号 E に対する生成規則すべてを示せ．

$G_1(N_1, T_1, P_1, S_1)$

- 非終端記号 (non-terminal symbol) の集合 $N_1 = \{ E, T, F, I \}$
- 終端記号 (terminal symbol) の集合 $T_1 = \{ a, b, (,), *, + \}$
- 生成規則 (generating rule) の集合 $P_1 = \{ \ E \to T, \quad E \to E + T, \quad T \to F, \quad T \to T * F,$
  $F \to I, \quad F \to (E), \quad I \to a, \quad I \to b, \quad I \to Ia, \quad I \to Ib \quad \}$
- 始記号 (start symbol) $S_1 = E$

(3) プログラミング言語の文法として，正規文法 (regular grammar) や文脈依存文法 (context-sensitive grammar) ではなく，文脈自由文法を用いる事のそれぞれに対する利点を簡潔に述べよ．

(4) オートマトン (automaton) に関する以下の各小問に答えよ．オートマトン M によって認識される言語 (language recognized by M) を $L(M)$ と表記する．すべてのオートマトンについて，入力記号 (input symbols) の集合を $\Sigma = \{0, 1\}$ とする．また，関数 $F : \Sigma^* \to \mathrm{N}$ を，任意の語 (word) $w = a_1 a_2 ... a_n \in \Sigma^*$ に対し，w を a_1 を最下位ビット (least significant bit) とした 2 進表現とみなして 0 以上の整数値を与える関数とする (N は 0 以上の整数の集合を表す)．ただし，空語 (empty word)ε に対しては 0 を与える．すなわち，a_i を整数値とみなして F を以下のように定義する．

$$F(w) = \begin{cases} \displaystyle\sum_{i=1}^{n} 2^{i-1} \cdot a_i & w = a_1 a_2 ... a_n, n \geq 1 \\ 0 & w = \varepsilon \end{cases}$$

たとえば以下が成り立つ．$F(\varepsilon) = 0, F(0) = 0, F(10) = 1, F(0001) = 8, F(1100) = 3$

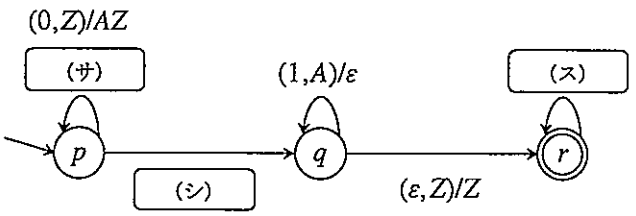
(4-1) $L(M_1) = \{w \mid w \in \Sigma^*, \exists k \in \mathrm{N} : F(w) = 3k\}$ である決定性有限オートマトン (deterministic finite automaton) M_1 について考える．以下は，M_1 の状態遷移表 (state transition table) である．

	0	1
→ ⓐ	b	d
ⓑ	(ア)	(イ)
c	(ウ)	(エ)
d	c	a
e	(オ)	(カ)
f	(キ)	(ク)

注．→ は初期状態，◯は受理状態を示す．

上の状態遷移表の空白部分 (ア)〜(ク) を埋めよ．ただし，M_1 の初期状態 (initial state) は a, 受理状態 (accepting states) は a と b である．

(4-2) 最終状態による受理 (acceptance by final state) を行うプッシュダウンオートマトン (pushdown automaton) M_2 を，$L(M_2) = \{w \mid w \in \Sigma^*, \exists k \in \mathrm{N} : (F(w) = 2^{2k} - 2^k$かつ $F(w) > 0)\}$ となるように構成することを考える．スタック記号 (stack symbols) は A と Z であり，Z を初期 (底) 記号 (initial stack symbol) とする．以下は，M_2 の状態遷移図 (state transition diagram) である．図の空白部分 (サ)，(シ)，(ス) を適切な動作で埋めよ．ただし，M_2 の初期状態 (initial state) は p, 受理状態 (accepting state) は r である．



配点: (1-1) 8 点, (1-2) 8 点, (1-3) 8 点, (1-4) 10 点, (1-5) 10 点,
(2-1) 30 点, (2-2) 12 点, (2-3) 24 点, (2-4) 15 点

(1) 通信路符号化に関する以下の文章を読み，その下の各小問に答えよ．

C を 2 元ブロック符号（binary block code）とし，正整数 d を符号 C の最小ハミング距離（minimum Hamming distance）とする．t を，$2t+1 \leq d$ を満たす任意の非負整数とする．受信語（received word）v に対する，限界距離（bounded distance）を t とした復号法（decoding algorithm）とは，以下のような手続きである．

(i) $C_t(v) = \{w \mid w \in C, \Delta(v,w) \leq t\}$ を求める．ここで $\Delta(v,w)$ は語 v と w の間のハミング距離（Hamming distance）を表す．このとき，$C_t(v)$ の要素数は高々1 となる(あ)．

(ii) $C_t(v)$ が空集合でなければ，その要素を v の復号結果とする．$C_t(v)$ が空集合ならば，［　（い）　］．

以下では，C は線形符号（linear code）であるとする．受信語（を表す行ベクトル）v と，検査行列（check matrix）H の転置行列（transposed matrix）H^{T} との積 vH^{T} は，［　（う）　］と呼ばれる．［　（う）　］は，v が符号語（codeword）のときかつそのときのみ零ベクトルとなる．さらに，ハミング距離 $(d-1)/2$ 以内に符号語 w が存在するような v に対しては，［　（う）　］は v と w の相違箇所のみに依存して定まる行ベクトルとなる．

(1-1) $C = \{000, 111\}$ の場合の $C_0(010)$ と $C_1(101)$ をそれぞれ求めよ．

(1-2) 限界距離復号法の正しい動作を表す文となるように，空欄（い）を埋めよ．

(1-3) 空欄（う）を，適切な用語で埋めよ．

(1-4) 任意の符号 C と任意の受信語 v について下線部（あ）が成立することを説明せよ．必要であれば，ハミング距離が三角不等式（triangle inequality）を満たすことを既知の事実として用いてよい．

(1-5) 符号 $C = \{000, 111\}$ は線形符号である．この符号 C の検査行列を一つ答えよ．どのようにして求めたかも記述すること．

(2) TCP/IP 参照モデル（reference model）におけるトランスポート層（transport layer）に関する以下の文章を読み，その下の各小問に答えよ．

> TCP/IP 参照モデルにおけるトランスポート層は，アプリケーション層（application layer）で動作する二つのプロセス（process）の間に論理的な通信路（channel）を提供する．論理的な通信路とは，物理的には接続されていないものの，アプリケーション層で動作するプロセスにとってはあたかも物理的に接続されているように見える通信路である．アプリケーション層で動作するプロセスは，トランスポート層により提供される論理的な通信路を用いてデータを転送する．
>
> TCP/IP 参照モデルにおけるトランスポート層プロトコル（transport layer protocol）の代表的なものとして，TCP と UDP がある．TCP は，シーケンス番号（sequence number），確認応答（ACK），タイマ等を用いて，データが欠損することなく正しい順序で転送されることを保証するコネクション型（connection-oriented）のプロトコルである．TCP では，データ転送の開始前にスリーウェイハンドシェイク（three-way handshake）を行い，プロセス間のコネクションを設定する．一方，UDP は，プロセス間のコネクションを設定することなくデータ転送を開始するコネクションレス型（connectionless）のプロトコルである．

以降では，トランスポート層のサービスを提供するハードウェア（hardware）もしくはソフトウェア（software）をトランスポートエンティティ（transport entity）と呼ぶ．

(2-1) トランスポート層プロトコルとして UDP を用いる場合の，TCP を用いる場合と比較しての利点を二つ述べよ．それぞれについて，その利点を有する理由も併せて述べよ．

(2-2) アプリケーション層で複数のプロセスが動作する場合，トランスポート層ではネットワーク層（network layer）から TPDU（Transport Protocol Data Unit）を受け取り，TPDU に格納されたデータを適切なプロセスに渡さなければならない．

トランスポートエンティティは，四つの情報の組み合わせによって TCP のコネクションを識別し，TCP の TPDU であるセグメント（segment）に格納されたデータを適切なプロセスに受け渡す．TCP のコネクションを識別するために必要な四つの情報を列挙せよ．

(2-3) 図 1 は，アプリケーション層で動作するプロセス A とプロセス B の間で TCP を用いてデータを転送する際の，トランスポートエンティティ間におけるセグメント送受信の例を示している．この例では，プロセス A からプロセス B に対してデータを転送している．

プロセス A 側のトランスポートエンティティは，プロセス A からコネクション設定を依頼されるとスリーウェイハンドシェイクを開始する．スリーウェイハンドシェイクでは，プロセス A 側のトランスポートエンティティおよびプロセス B 側のトランスポートエンティティがそれぞれランダムに初期シーケンス番号（initial sequence number）を決定し，初期シーケンス番号を互いに通知する．

トランスポートエンティティが，プロセスからのコネクション設定依頼に対して常に同じ初期シーケンス番号を用いる場合に生じる問題を二つ述べよ．それぞれの問題について，その問題がどのような状況で生じるかを説明し，スリーウェイハンドシェイクにおいて初期シーケンス番号をランダムに決定することにより問題がどのように回避されるかを述べよ．

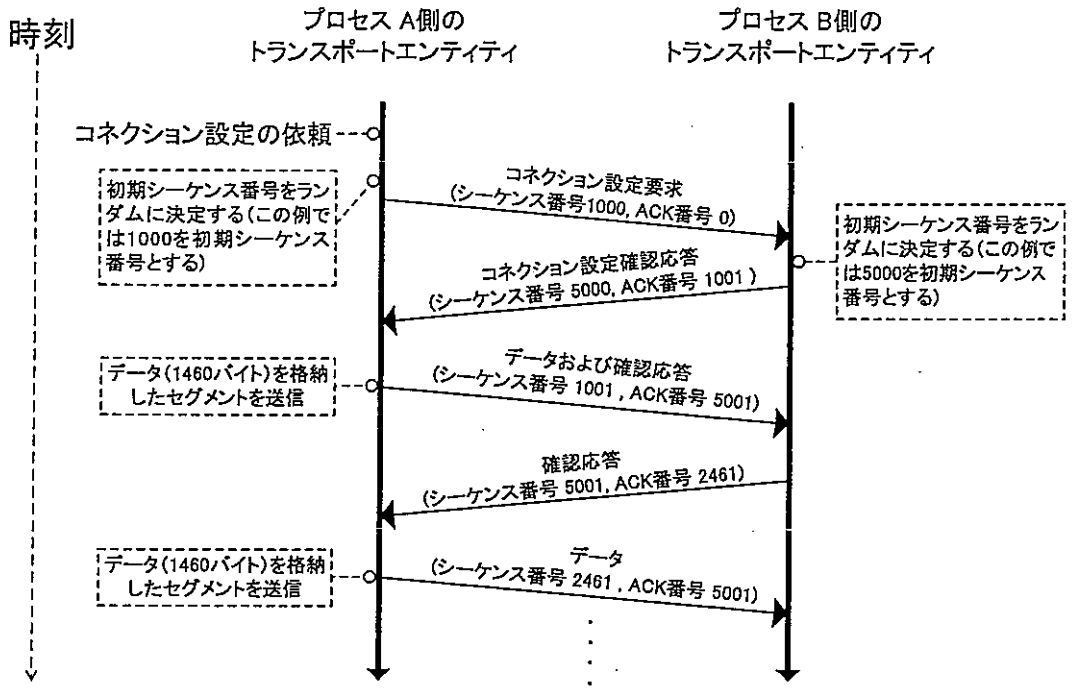


図 1

(2-4) プロセス A が TCP を用いてプロセス B から 100000 [バイト] のファイル（file）を取得する場合を考える．以下に示した条件において，プロセス A 側のトランスポートエンティティが送信するセグメントの総数と，プロセス B 側のトランスポートエンティティが送信するセグメントの総数を，それぞれ答えよ．計算過程も示せ．なお，コネクション解放時に送信されるセグメントの数は総数に含めない．

- プロセス A とプロセス B の間にコネクションを新たに設定し，そのコネクションを用いてデータを転送する．また，プロセス A 側のトランスポートエンティティがスリーウェイハンドシェイクを開始する．
- プロセス A は，プロセス B に対して取得するファイルのファイル名等が記述された 100 [バイト] のデータを転送する．また，プロセス B は，プロセス A に対して取得するファイルのサイズ等が記述された 100 [バイト] のデータと取得するファイルからなる 100100 [バイト] のデータを転送する．
- セグメントのヘッダ（header）のサイズを 20 [バイト]，ヘッダを含めた最大セグメントサイズ（maximum segment size）を 1000 [バイト] とする．
- トランスポートエンティティは，データが格納されたセグメントを一つ受信すると確認応答のセグメントを一つ送信する．
- トランスポートエンティティは，確認応答のセグメントをピギーバック（piggyback）する．すなわち，トランスポートエンティティは，転送するデータがある場合，確認応答のセグメントにデータを格納する．
- トランスポートエンティティがセグメントの送受信に用いるバッファは十分に大きく，常に最大セグメントサイズ以上の空き領域がある．
- ファイルやデータの圧縮は考えない．
- セグメントの損失は生じない．

配点: (1-1) 30 点, (1-2) 40 点, (1-3) 30 点, (2) 25 点

連続する 4 個以上の 1 または連続する 3 個以上の 0 を検出する順序回路(sequential circuit)を設計する. この回路は 1 ビットの入力(input) x と 1 ビットの出力(output) z を持つ. 各時刻には, x に 0 または 1 が入力される. x に 1 が 4 個以上連続入力される, あるいは x に 0 が 3 個以上連続入力されると z は 1 を出力し, それ以外の場合は 0 を出力する. 例えば, x に 0110111110100001 が入力された場合の出力は表 1 のようになる. ただし時刻 0 以前に入力はない.

表 1

時刻		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	...
入力	x	0	1	1	0	1	1	1	1	1	0	1	0	0	0	0	1	...
出力	z	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	...

(1) この回路を 6 状態(state) A, B, C, D, E, F を持つミーリー(Mealy)型順序回路として設計する. 以下の小問に答えよ.

(1-1) 各状態を表 2 のように定めるとして, 状態遷移図 (state transition diagram) を示せ. 表中で現時刻は T とする. 出力 z も図中に記載せよ.

表 2

A	初期状態(過去に何も入力されていない状態)
B	時刻 T-1 に 0 が入力されたが, 時刻 T-2 には 0 が入力されていない状態
C	時刻 T-1, T-2 に共に 0 が入力された状態
D	時刻 T-1 に 1 が入力されたが, 時刻 T-2 には 1 が入力されていない状態
E	時刻 T-1, T-2 に共に 1 が入力されたが, 時刻 T-3 には 1 が入力されていない状態
F	時刻 T-1, T-2, および T-3 にすべて 1 が入力された状態

(1-2) 表 3 のように 3 ビットを用いた状態割り当て(state assignment)を行い, 3 個の D フリップフロップ(flip flop)を用いて設計する．状態変数(state variable) q_1, q_2, q_3 に対応するフリップフロップの D 入力を d_1, d_2, d_3 とする．d_1, d_2, d_3 および z を q_1, q_2, q_3, x の最小積和形(最簡積和形, minimum sum-of-products expression)で表せ．

表 3

状態	q_1	q_2	q_3
A	0	0	0
B	0	0	1
C	0	1	1
D	1	1	1
E	1	1	0
F	1	0	0

(1-3) (1-2)で求めた最小積和形を利用して， D フリップフロップと NAND, NOT ゲートを用いてこの回路を実現し，その回路図を示せ．ただし、NAND ゲートの最大入力数は 3 とする．

(2) 2 入力 NAND ゲートを PMOS, NMOS からなる CMOS 回路として実現したときの回路図を示せ．電源を Vdd, グラウンドを Vss, 入力信号をそれぞれ IN_1, IN_2, 出力を OUT として記載せよ．

【選択問題】数学解析と信号処理

配点：(1) 40 点，(2) 45 点，(3) 40 点

以下の各問に答えよ．

(1) 常微分方程式 (ordinary differential equation) $f''(t) + 3f'(t) + 2f(t) = 0$ を，ラプラス変換 (Laplace transform) を用いて，初期条件 (initial condition) $f(0) = 2, f'(0) = -3$ のもとで解け．

(2) 以下の定積分 (definite integral) I を留数定理 (residue theorem) を用いて求めよ．

$$I = \int_0^{2\pi} \frac{1}{5 + 4\sin\theta} d\theta$$

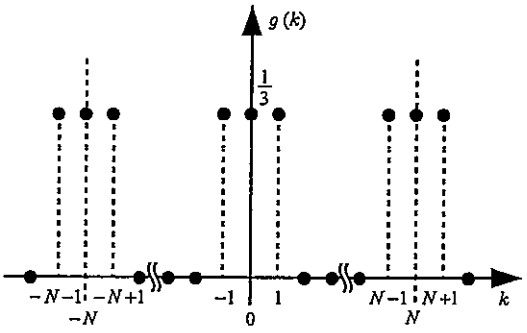
(次ページへ続く)

(3) 整数 (integer) k, n に対し，周期 (period) N の周期的 (periodic) な離散信号 (discrete signal) $f(k)$ を考え，$f(k)$ の離散フーリエ変換 (discrete Fourier transform) $F(n)$ を

$$F(n) = \sum_{k=0}^{N-1} f(k) e^{-j\frac{2\pi}{N}nk} \tag{1}$$

と定義する．なお，ここで j は虚数単位 (imaginary unit) を表わす．また，入力 (input) $f(k)$ に対し，$h(k) = \{f(k-1) + f(k) + f(k+1)\}/3$ を出力 (output) するフィルタ (filter) を考える．このとき，以下の文章の空欄 (A), (B) を適切な語句または数式で埋めよ．また，空欄 (c)〜(h) に当てはまる最も適切な語句または数式を下記に示す選択肢から選んで記号を解答せよ．

フィルタの出力信号 $h(k)$ は，入力信号 $f(k)$ および下図に示す関数 $g(k)$ を用いて，$h(k) =$ [(c)] と表わされる．$g(k)$ の離散フーリエ変換 $G(n)$ を求めると，$G(n) =$ [(A)] となる．$h(k)$ の離散フーリエ変換 $H(n)$ は，$F(n), G(n)$ を用いて $H(n) =$ [(d)] と表わされる．$|F(n)|^2$ や $|H(n)|^2$ は信号の [(B)] と呼ばれ，信号が運ぶ周波数毎のエネルギーを表している．上述の離散フーリエ変換の定義に対応する逆離散フーリエ変換 (inverse discrete Fourier transform) は，$f(k) =$ [(e)] である．



このフィルタは，[(f)] フィルタであり，信号の [(g)] する効果がある．

離散フーリエ変換を全ての $n = 0, 1, \cdots, N-1$ について計算機で数値計算する場合，式 (1) の通りに計算すると，時間計算量 (time complexity) は O([(h)]) となる．

【選択肢】

(ア) アナログ (analog),
(イ) ハイパス (high-pass),
(ウ) ローパス (low-pass),
(エ) バンドパス (band-pass),
(オ) バンドストップ (band-stop),
(カ) マルチバンド (multi-band),
(キ) 振幅 (amplitude) を増幅 (amplify),
(ク) コントラスト (contrast) を強調 (enhance),
(ケ) 分解能 (resolution) を改善 (improve),
(コ) 雑音 (noise) を低減 (reduce),
(サ) エッジ (edge) を鮮鋭に (sharpen),
(シ) $f(k)g(k)$,
(ス) $F(n)G(n)$,
(セ) $\sum_{m=0}^{k} f(m)g(m)$,
(ソ) $\sum_{m=0}^{n} F(m)G(m)$,
(タ) $\sum_{m=0}^{N-1} f(m)g(k-m)$,
(チ) $\sum_{m=0}^{N-1} F(m)G(n-m)$,
(ツ) $\sum_{n=0}^{N-1} F(n)e^{j\frac{2\pi}{N}nk}$,
(テ) $\frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} F(n)e^{j\frac{2\pi}{N}nk}$,
(ト) $\frac{1}{N} \sum_{n=0}^{N-1} F(n)e^{j\frac{2\pi}{N}nk}$,
(ナ) $\int_{-\infty}^{\infty} f(x)g(k-x)dx$,
(ニ) $\int_{-\infty}^{\infty} F(\omega)G(n-\omega)d\omega$,
(ヌ) $\frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega)e^{-j\omega k}d\omega$,
(ネ) N,
(ノ) $N \log N$,
(ハ) $N\sqrt{N}$,
(ヒ) N^2