令 和 4 年 度

名古屋大学大学院情報学研究科
情報システム学専攻
入学試験問題（専門）

令和３年８月５日

注意事項

１．試験開始の合図があるまでは、この問題冊子を開いてはならない。

２．試験終了まで退出できない。

３．外国人留学生の志願者は、日本語と日本語以外の１言語間の辞書１冊に限り使用してよい。電子辞書の持ち込みは認めない。

４．日本語または英語で解答すること。

５．問題冊子、解答用紙５枚、草稿用紙５枚が配布されていることを確認すること。

６．問題は問１～７の７問がある。このうち５問を選択して解答すること。なお、選択した問題番号を解答用紙の指定欄に記入すること。

７．全ての解答用紙の所定の欄に受験番号を必ず記入すること。解答用紙に受験者の氏名を記入してはならない。

８．解答用紙に書ききれない場合は、裏面を使用してもよい。ただし、裏面を使用した場合は、その旨、解答用紙表面右下に明記すること。

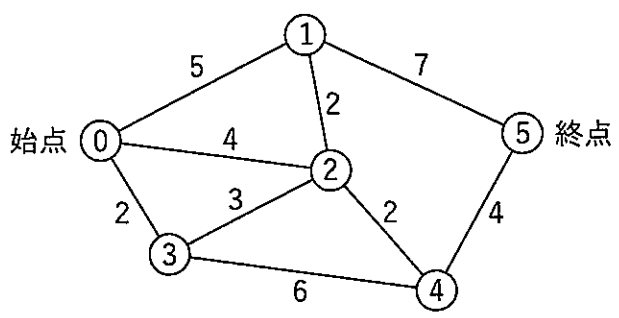
９．解答用紙は試験終了後に５枚とも提出すること。

１０．問題冊子、草稿用紙は試験終了後に持ち帰ること。

問1

頂点と辺からなり，各辺に距離が与えられている無向グラフに対して，頂点間の最短経路を求めるプログラムを考える。

プログラム1は，下図に示すグラフの始点から終点までの最短経路とその距離を出力するC言語プログラムである。グラフの構造は，配列 graph_data に格納されている。配列の各要素が各辺に対応しており，要素を構成する3つの数値は，辺の両端の頂点と距離を表している。なお，プログラムの行頭の数字は行番号を表す。



このプログラムを実行すると，次のような出力が得られる。

```
total distance: 10
shortest path: 0 -> 2 -> 4 -> 5
```

このプログラムに関して，以下の問いに答えよ。

(1) [A], [B], [C] を埋めて，プログラムを完成させよ。

(2) このプログラムを実行すると，61行目の代入文は何回実行されるか答えよ。

(3) このプログラムを次のように変更した場合の出力を答えよ。なお，それぞれの変更は個別に行うものとする。

  (a) 16行目の { 0, 2, 4 } を，{ 0, 2, 6 } に変更した場合

  (b) 20行目の { 2, 4, 2 } を，{ 2, 4, 5 } に変更した場合

(4) このプログラムが “error!” を出力するのは，与えるグラフ（graph_data の値）がどのような構造になっている場合かを答えよ。

Translation of technical terms

頂点	node	終点	end node
辺	edge	配列	array
距離	distance	要素	element
無向グラフ	undirected graph	行番号	line number
最短経路	shortest path	代入文	assignment statement
始点	start node		

プログラム1

```
 1    #include <stdio.h>
 2
 3    typedef struct {
 4      int   node1;
 5      int   node2;
 6      int   distance;
 7    } EDGE;
 8
 9    #define NUM_NODE    6
10    #define NUM_EDGE    9
11    #define START_NODE  0
12    #define END_NODE    5
13
14    const EDGE graph_data[NUM_EDGE] = {
15      { 0, 1, 5 },
16      { 0, 2, 4 },
17      { 0, 3, 2 },
18      { 1, 2, 2 },
19      { 2, 3, 3 },
20      { 2, 4, 2 },
21      { 3, 4, 6 },
22      { 4, 5, 4 },
23      { 1, 5, 7 },
24    };
25
26    typedef struct {
27      int   total_distance;
28      int   previous_node;
29    } NODE;
30
31    NODE node_stat[NUM_NODE];
32
33    #define UNKNOWN         -1
34    #define LARGE_DISTANCE  10000
35
36    void print_path(int node)
37    {
38      if (node != START_NODE) {
39        [      A      ];
40        printf(" -> %d", node);
41      }
42      else {
43        printf("shortest path: %d", node);
44      }
45    }
46
47    int main()
48    {
49      int   i, j;
50      int   min_src_node, min_dst_node, min_distance;
51      int   node1, node2, distance, new_distance;
52
```

```
53      for (i = 0; i < NUM_NODE; i++) {
54        node_stat[i].total_distance = UNKNOWN;
55      }
56      node_stat[START_NODE].total_distance = 0;
57
58      while (1) {
59        min_distance = LARGE_DISTANCE;
60        for (j = 0; j < NUM_EDGE; j++) {
61          node1 = graph_data[j].node1;
62          node2 = graph_data[j].node2;
63          distance = graph_data[j].distance;
64
65          if (node_stat[node1].total_distance != UNKNOWN) {
66            if (node_stat[node2].total_distance == UNKNOWN) {
67              new_distance = node_stat[node1].total_distance + distance;
68              if (new_distance < min_distance) {
69                min_distance = new_distance;
70                min_src_node = node1;
71                min_dst_node = node2;
72              }
73            }
74          }
75          else {
76            if (node_stat[node2].total_distance != UNKNOWN) {
77              new_distance = node_stat[node2].total_distance + distance;
78              if (new_distance < min_distance) {
79                min_distance = new_distance;
80                min_src_node = [  B  ];
81                min_dst_node = [  C  ];
82              }
83            }
84          }
85        }
86        if (min_distance == LARGE_DISTANCE) break;
87        node_stat[min_dst_node].total_distance = min_distance;
88        node_stat[min_dst_node].previous_node = min_src_node;
89      }
90
91      if (node_stat[END_NODE].total_distance == UNKNOWN) {
92        printf("error!");
93      }
94      else {
95        printf("total distance: %d\n", node_stat[END_NODE].total_distance);
96        print_path(END_NODE);
97      }
98      printf("\n");
99    }
```

問2

本問では，アルファベットを$T = \{0, 1\}$とする．記号列に関する次の2つの条件を考える．

(C1) 長さが2以上であり，左から2番目の記号が1である．
(C2) 長さが3以上であり，右から3番目の記号が1である．

(1) 以下のような決定性有限オートマトン (以下，DFA と略す) の状態遷移図をそれぞれ示せ．初期状態，受理状態を明記すること．

(a) 条件 (C1) を満たす記号列かつそれらのみを受理する状態数4以下の DFA A_1

(b) 条件 (C2) を満たす記号列かつそれらのみを受理する状態数8以下の DFA A_2

(2) 条件 (C1) を満たさないか，または，条件 (C2) を満たす記号列全部からなる言語をL_3とする．

(a) L_3 に属する長さ3以下の記号列をすべて書け．

(b) L_3 に属する長さ4の記号列は何個あるか．150字程度（英語の場合，100語程度）の理由とともに答えよ．

(3) L_3 を受理する DFA の状態遷移図を示せ．

(4) 任意に与えられた DFA A に対し，「A の受理する記号列はすべて条件 (C2) を満たす」かどうかは一定の手順（アルゴリズム）で判定できることを説明せよ．次の (性質1) を利用してもよい．

(性質1) 任意に与えられた DFA B に対し，B の受理する言語が空集合であるかどうかを判定するアルゴリズムが存在する．

Translation of technical terms

アルファベット	alphabet
記号列	string
決定性有限オートマトン	deterministic finite automaton
状態遷移図	state transition diagram
初期状態	initial state
受理状態	accepting state
言語	language
アルゴリズム	algorithm
判定する	decide
空集合	empty set

問3

自然数の有限列のうち，含まれている要素が重複しないものを考える．そのような列 S に含まれる要素のうちで k 番目に小さい数（小さい方から数えて k 番目の数）を求める再帰的アルゴリズム select（図 1）について以下の問いに答えよ．なお，$|X|$ は有限列 X の要素数を表し，$\lceil\ \rceil$ は天井関数を表す．天井関数は引数として受け取る実数 r に対して，r 以上の最小の整数を返す．さらに，有限列 X の中央値は X の中で $\lceil \frac{|X|}{2} \rceil$ 番目に小さい要素とする．

(1) S を以下の列，k を 10 とする．

$$11, 12, 16, 33, 2, 18, 39, 15, 21, 7, 37, 29, 40, 6, 25, 27, 14, 4, 35, 28, 22, 20, 17, 3, 1$$

(a) このような S，k に対して図 1 の手順の 1. から 5. を順に実行したときに求められる手順の中の列 M，自然数 x，列 A を示せ．なお，列 A の要素の並び順についてはアルゴリズムでは言及していないので，解答では並び順を問わない．

(b) このような S，k に対して select(S, k) を実行したときの出力を答えよ．

(2) $n = |S|$ とし，n を 10 の倍数としたときに列 A に含まれ得る要素の数の最小値と最大値それぞれを n を用いた式で表せ．

(3) $n = |S|$ とする．図 1 の手順において，2. における列 $G_1, \ldots, G_N$ を得る操作，5. における列 A, B を得る操作の最大時間計算量をそれぞれ $O(n)$ とする．このとき，select(S, k)（ただし，$0 < k \leq n$）を実行したときの最大時間計算量をオーダー記法で示せ．解答では $|S|$ ではなく n を用いること．

(4) 以下の整列アルゴリズムを考える．

(a) バブルソート　(b) マージソート　(c) ヒープソート

(d) クイックソート　(e) 挿入ソート　(f) バケットソート

(a)〜(f) のうち，入力として与えられる列の要素数を n としたときに以下の条件をすべて満たすものを 1 つ選択せよ．

- 最大時間計算量が $O(n^2)$ である．
- 図 1 のアルゴリズムを利用することで最大時間計算量を $O(n \log_2 n)$ に改善できる．

再帰的アルゴリズム select

入力 重複する要素を持たない自然数の有限列 S，正整数 k（ただし，$k \leq |S|$）

出力 列 S に含まれる要素のうちで k 番目に小さいもの

手順

1. N を $\lceil \frac{|S|}{5} \rceil$ とする．
2. 列 S を先頭から順に要素 5 つずつの列 $G_1, \ldots, G_N$ に分ける．つまり，列 G_1 から列 G_N を順に連結(れんけつ)すると列 S に一致する．さらに，列 $G_1, \ldots, G_{N-1}$ それぞれに含まれる要素数は 5 であり，$|S|$ が 5 の倍数ではないときに列 G_N の要素数は 5 に満たない．
3. M を列 $G_1, \ldots, G_N$ それぞれの中央値からなる列とする．なお，列 M の要素の並び順は先頭から順に列 $G_1, \ldots, G_N$ の中央値が並んでいるとする．
4. x を列 M の中央値，すなわち，$\mathrm{select}(M, \lceil \frac{|M|}{2} \rceil)$ の返(かえ)り値(ち)とする．
5. 列 A, B を以下を満たす列とする．

$$\{a \mid a \text{は列 } A \text{ に含まれる要素}\} = \{y \in S \mid y < x\}$$

$$\{b \mid b \text{は列 } B \text{ に含まれる要素}\} = \{y \in S \mid x < y\}$$

6. $|A| = k - 1$ のときは x を返して終了し，そうでないときは次に進む．
7. $|A| \geq k$ のときは $\mathrm{select}(A, k)$ の返り値を返して終了し，そうでないときは次に進む．
8. $\mathrm{select}(B, k - |A| - 1)$ の返り値を返して終了する．

図 1: k 番目に小さい要素を求めるアルゴリズム

Translation of technical terms

列	sequence	整列アルゴリズム	sorting algorithm
再帰的アルゴリズム	recursive algorithm	バブルソート	bubble sort
要素数	cardinality	マージソート	merge sort
天井関数	ceiling function	ヒープソート	heap sort
引数	argument	クイックソート	quick sort
中央値	median	挿入ソート	insertion sort
手順	procedure	バケットソート	bucket sort
最大時間計算量	worst-case time complexity	連結する	append
		返り値	return value
オーダー記法	big O notation		

問4

一階述語論理において，1引数の関数記号 f と，2引数の述語記号 R からなるシグネチャ Σ 上の論理式を考える．論理結合子として，$\wedge$（かつ，連言），$\to$（ならば，含意），$\neg$（でない，否定），量化子として，$\forall$（全称量化子），$\exists$（存在量化子）を用いる．x，y，z によって項変数を表すとする．論理式の解釈は古典論理の真偽値意味論によるものとする．$\mathbb{N}$ は0以上の自然数の集合とする．

(1) 以下の論理式 (a)〜(f) のうち，トートロジーであるものを全て挙げよ．

(a)	$\forall x R(x, f(x))$	(d)	$(\forall x \forall y R(x, y)) \to \forall x R(x, x)$
(b)	$\exists x R(f(x), x)$	(e)	$\forall x \forall y (R(x, y) \to R(f(x), f(y)))$
(c)	$\exists x \forall y R(y, x)$	(f)	$(\neg \forall x \exists y R(x, f(y))) \to \exists x \neg \exists y R(x, f(y))$

(2) 領域が $\mathbb{N}$ であるような Σ 上の構造 $\mathcal{M}_1 = \langle \mathbb{N}; f^{\mathcal{M}_1}; R^{\mathcal{M}_1} \rangle$ を考える．$\mathcal{M}_1$ における f の解釈 $f^{\mathcal{M}_1}$ と R の解釈 $R^{\mathcal{M}_1}$ は以下のように定義されているとする．

$$f^{\mathcal{M}_1}(n) = 2 \times n \qquad\qquad R^{\mathcal{M}_1}(n, m) \text{ は真} \overset{\text{def}}{\Leftrightarrow} n \leq m$$

(1) の論理式 (a)〜(f) のうち，構造 $\mathcal{M}_1$ における解釈が真となるものを全て挙げよ．

(3) 3つの論理式

$$A = \forall x (\neg R(x, x)) \quad B = \forall x \forall y \forall z ((R(x, y) \wedge R(y, z)) \to R(x, z)) \quad C = \forall x R(f(x), x)$$

の解釈が全て真となる構造 $\mathcal{M}_2 = \langle \mathbb{N}; f^{\mathcal{M}_2}; R^{\mathcal{M}_2} \rangle$ を一つ示せ．

(4) (3) の A と B，および，以下の2つの論理式

$$D = \forall x \forall y (R(x, y) \to \exists z (R(x, z) \wedge R(z, y))) \qquad E = \forall x \exists y R(x, y)$$

の解釈が全て真となる構造 $\mathcal{M}_3 = \langle \mathbb{N}; f^{\mathcal{M}_3}; R^{\mathcal{M}_3} \rangle$ を一つ示せ．ただし，論理式 A，B，D，E には関数記号 f が出現しないので，$R^{\mathcal{M}_3}$ のみ示せばよい．

Translation of technical terms

一階述語論理	first-order logic	全称量化子	universal quantifier
1引数	unary	存在量化子	existential quantifier
関数記号	function symbol	項変数	term variable
2引数	binary	解釈	interpretation
述語記号	predicate symbol	古典論理	classical logic
シグネチャ	signature	真偽値意味論	truth value semantics
論理式	formula	自然数	natural number
論理結合子	logical connetive	集合	set
連言	conjunction	トートロジー	tautology
含意	implication	領域	domain
否定	negation	構造	structure
量化子	quantifier	真	true

問5

構文解析に関する以下の問いに答えよ．ここで文脈自由文法Gは，非終端記号の集合N，終端記号の集合T，生成規則の集合P，開始記号Sからなる4つ組(N, T, P, S)で表されるとする．

(1) 生成規則の集合P_1を以下に示す．文脈自由文法$G_1 = (\{E\}, \{+, *, (,), i\}, P_1, E)$は，文$i + i * i$に対して2通りの最左導出を行うことができる．それら最左導出の解析木を示せ．

$$\begin{aligned} P_1 = \{\ E &\rightarrow E + E, \\ E &\rightarrow E * E, \\ E &\rightarrow (E) \mid i\ \} \end{aligned}$$

(2) 生成規則の集合P_2が以下のように与えられ，$FIRST(\alpha)$がαから導出される記号列の先頭に現れる終端記号の集合を表すとき，文脈自由文法$G_2 = (\{E, F, T\}, \{\vee, \wedge, \neg, (,), i, j\}, P_2, E)$に対して，$FIRST(E)$を求めよ．

$$\begin{aligned} P_2 = \{\ E &\rightarrow E \vee F \mid F, \\ F &\rightarrow F \wedge T \mid T, \\ T &\rightarrow \neg T \mid (T) \mid i \mid j\ \} \end{aligned}$$

(3) 生成規則の集合P_3を以下に示す．文脈自由文法$G_3 = (\{E, F, T\}, \{+, *, (,), i\}, P_3, E)$は左再帰性があるため，このまま下向き構文解析による最左導出を行うと問題が生じる．下向き構文解析における左再帰性の問題について，100文字（英文の場合は50語）以内で説明せよ．

$$\begin{aligned} P_3 = \{\ E &\rightarrow E + T \mid T, \\ T &\rightarrow T * F \mid F, \\ F &\rightarrow (E) \mid i\ \} \end{aligned}$$

(4) G_3を左再帰性のない等価な文脈自由文法$G_4 = (\{E, E', F, T, T'\}, \{+, *, (,), i\}, P_4, E)$に書き換えることを考える．以下の（ア）と（イ）を埋め，G_3とG_4が等価になるようにP_4を完成させよ．

$$\begin{aligned} P_4 = \{\ E &\rightarrow TE', \\ E' &\rightarrow \boxed{\text{（ア）}}, \\ T &\rightarrow FT', \\ T' &\rightarrow \boxed{\text{（イ）}}, \\ F &\rightarrow (E) \mid i\ \} \end{aligned}$$

(5) 文脈自由文法 (N, T, P, S) において，$A \to \alpha$ および $A \to \beta$ が P に含まれ，$\alpha \neq \beta$ であるとする．このとき，(N, T, P, S) が LL(1)文法であるための必要条件として正しいものを，以下の (a)〜(c) から全て選べ．

(a) $FIRST(\alpha) = FIRST(\beta)$

(b) $FIRST(\alpha) \cap FIRST(\beta) = \varnothing$

(c) $FIRST(\alpha) \cup FIRST(\beta) = T$

Translation of technical terms:

構文解析	parsing	文	statement
文脈自由文法	context-free grammar	最左導出	leftmost derivation
非終端記号	nonterminal symbol	解析木	parse tree
集合	set	記号列	string
終端記号	terminal symbol	左再帰性	left recursion
生成規則	production rule	下向き構文解析	top-down parsing
開始記号	start symbol	文法	grammar
4 つ組	4-tuple	必要条件	necessary condition

問6

プロセッサがデータ・キャッシュを介して表1に示す主記憶のアドレスに上から順にアクセスする．アクセスの種類にはLoadとStoreがあり，いずれの種類のアクセスも4バイト単位で行われる．表1に示す“書き込みデータ”は，Storeアクセスによって主記憶に書き込もうとするデータを表す．表1に示すアクセス以外のメモリ・アクセスは発生せず，割込みも発生しないものとする．データ・キャッシュは当初は空であり，主記憶に格納されているデータは当初は全てゼロであるとする．以下の問いに答えよ．

番号	アドレス	アクセスの種類	書き込みデータ
1	0x0004	Load	—
2	0x0008	Store	0x00000001
3	0x000c	Store	0x00000002
4	0x0044	Load	—
5	0x0048	Store	0x00000003
6	0x0104	Load	—
7	0x0108	Store	0x00000004
8	0x0004	Store	0x00000005
9	0x0008	Store	0x00000006
10	0x000c	Store	0x00000007
11	0x0100	Load	—
12	0x010c	Store	0x00000008

表 1: メモリ・アクセス系列

(1) データ・キャッシュにおける有効ビットとダーティ・ビットの役割を合計200字（英語の場合90語）以内でそれぞれ説明せよ．

(2) 上記のデータ・キャッシュが，ブロック・サイズが16バイトで総容量が128バイトのダイレクト・マップ・キャッシュであり，書き込み方式がライト・スルー方式であるとき，次の(a)，(b)に答えよ．なお，上記のデータ・キャッシュは，Storeアクセスによるキャッシュのミス・ヒットが発生した場合にキャッシュ内にブロックを割り当てるライト・アロケート方式を採用しているものとする．

(a) 表1に示す12回のメモリ・アクセスにともなうキャッシュ・アクセスがそれぞれヒットするかミス・ヒットするか，その理由とあわせて答えよ．表2の解答書式例を参考に解答表を作成し，12回全てのメモリ・アクセスに対してヒットまたはミス・ヒットとその理由を解答せよ．ミス・ヒットの理由には，初期参照ミスや過去のアクセスとのキャッシュ・ブロックの競合による競合性ミスなどがあり，ヒットの理由には，過去のアクセスと同一のキャッシュ・ブロックへのアクセスなどがある．競合性ミスやヒットの理由を解答する場合には，表2の例にならって必ずその原因となる過去のアクセスの番号を明示せよ．

(b) 表1に示すメモリ・アクセスが全て実行された直後に主記憶のアドレス0x0004，0x0008，0x000cにそれぞれ格納されているデータとそれらのデータが主記憶に格納されるまでの経緯を明示せよ．

(3) 上記のデータ・キャッシュが，ブロック・サイズが8バイトで総容量が128バイトの2ウェイ・セット・アソシアティブ・キャッシュであり，書き込み方式がライト・バック方式，ブロックの置き換え方式がLRU (Least Recently Used) であるとき，次の(a)，(b)に答えよ．なお，ライト・バック方式では，キャッシュのミス・ヒットが発生したときにのみ主記憶へのデータの書き戻しが行われる．

(a) 表1に示す12回のメモリ・アクセスにともなうキャッシュ・アクセスがそれぞれヒットするかミス・ヒットするか，上記の問題(2)(a)と同様の方法で答えよ．

(b) 表1に示すメモリ・アクセスが全て実行された直後に主記憶のアドレス0x0004，0x0008，0x000cにそれぞれ格納されているデータとそれらのデータが主記憶に格納されるまでの経緯を明示せよ．

番号	アドレス	ヒットまたはミス・ヒット	ヒットまたはミス・ヒットの理由
1	0x0004	.....	.....
...	.....	ミス・ヒット	i 番のアクセスとの競合性ミス
...	.....	ヒット	k 番のアクセスと同一ブロック
12	0x010c	.....	.....

表 2: 解答書式例

Translation of technical terms

プロセッサ	processor
データ・キャッシュ	data cache
主記憶	main memory
アドレス	address
アクセス	access
バイト	byte
メモリ・アクセス	memory access
割込み	interrupt
有効ビット	valid bit
ダーティ・ビット	dirty bit
ブロック・サイズ	block size
ダイレクト・マップ・キャッシュ	direct mapped cache
ライト・スルー	write through
ミス・ヒット	miss hit
ブロック	block
ライト・アロケート	write allocate
キャッシュ・アクセス	cache access
ヒット	hit
初期参照ミス	compulsory miss
競合性ミス	conflict miss
キャッシュ・ブロック	cache block
2 ウェイ・セット・アソシアティブ・キャッシュ	2-way set associative cache
ライト・バック	write back
置き換え方式	replacement policy

問 7

(1) 構造化分析設計法で使う DFD(Data Flow Diagram)に関して，図 1 の DFD の説明として正しいものをすべて選べ．なお，図 1 の矢印に付記した数字 1～5 はデータのラベル名を表し，円に付記したアルファベット A~D はプロセス名を表す．

① プロセス A からの出力データ 1 はプロセス D への入力データ 5 と同一である．

② プロセス B への入力データはない．

③ プロセス B の実行終了後でないとプロセス D は実行開始できない．

④ プロセス A の実行開始はプロセス C の実行開始前でなければならない．

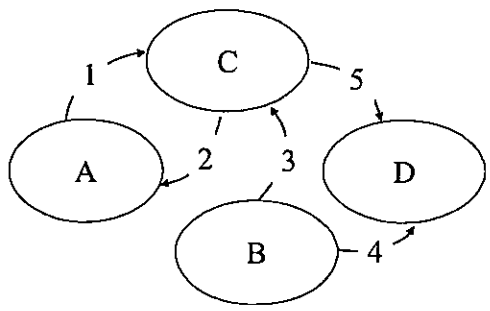


図 1 DFD

(2) UML(Unified Modeling Language)に関する下の問いに答えよ．なお，図 2 において，Table クラスの操作 get_value(k)は，引数で与えられた k と同じ値を属性 key として持つ KeyValue クラスのインスタンス keyValue_k を返す操作である．

(a) 図 2 のクラス図 C を汎化を用いないクラス図 C'に書き直せ．なお，C と C'でクラスの属性名と操作名は同一となるようにせよ．

(b) クラス図 C をオブジェクト指向プログラミング言語で実装したプログラムのソースコードを S とする．同様に，C'のソースコードを S'とする．このとき，KeyValue クラスの属性 key の値の一覧を返す操作 get_keys()を Table クラスと Hashtable クラスに追加することを考える．ソースコード S と S'の変更手順(変更方法)を合計 350 文字(英文の場合 160 語)以内で書け．なお，オブジェクト指向プログラミング言語は継承機能を持つものとし，使える場合には継承機能を使うこととする．

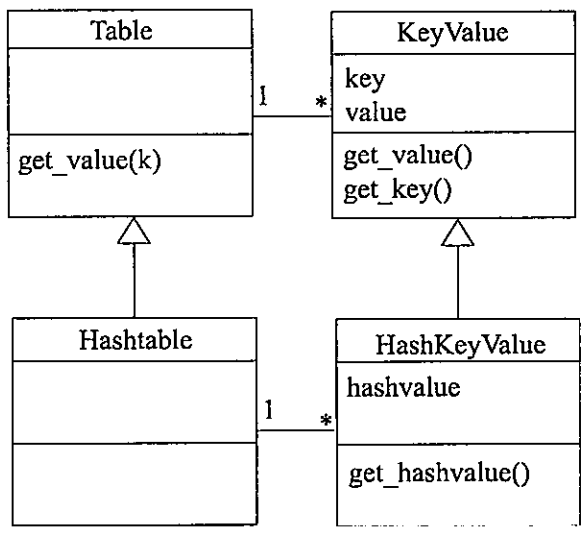


図 2 クラス図 C

(c) 次の振る舞いを表すシーケンス図を書け．なお，Table クラスと KeyValue クラスは図 2 のクラス図 C の Table と KeyValue を指すものとする．活性区間は省略すること．操作の呼出しはすべて同期呼出しとする．

> Search クラスは Table クラスの操作 get_value(k)により，k を属性 key に持つ KeyValue クラスのインスタンス keyValue_k を取り出し，その属性 value の値を操作 get_value()により取り出す．

Translation of technical terms

構造化分析設計法	structured analysis and design technique	操作名	operation name
ラベル	label	オブジェクト指向プログラミング言語	object-oriented programming language
プロセス	process		
データ	data	実装	implementation
クラス	class	プログラム	program
操作	operation	ソースコード	source code
引数	argument	継承機能	inheritance feature
属性	attribute	振る舞い	behavior
クラス図	class diagram	シーケンス図	sequence diagram
汎化	generalization	活性区間	activation bar
属性名	attribute name	同期呼出し	synchronous invocation