平成３１年度

筑波大学大学院博士課程
システム情報工学研究科
コンピュータサイエンス専攻

博士前期課程（一般入学試験　８月期）
試験問題　基礎科目（数学，情報基礎）
Mathematics／Fundamentals of Computer Science

[注意事項][Instructions]

1. 試験開始の合図があるまで，問題の中を見てはいけません．また，筆記用具を手に持ってはいけません．
   Do NOT open this booklet before the examination starts. In addition, do not have a pen in hand before the examination starts.
2. 試験開始の合図のあとで，全ての解答用紙の定められた欄に，研究科，専攻，受験番号を記入すること．
   Fill in the designated spaces on each answer sheet with the name of the graduate school, name of your main field, and the examination number after the examination starts.
3. この問題は全部で 15 ページ（表紙を除く）です．1〜7 ページは日本語版，8〜15 ページは英語版です．
   This booklet consists of 15 pages, excluding the cover sheet. The Japanese version is shown on pages 1–7 and the English version on pages 8–15.
4. 解答用紙（罫線有り）を３枚，下書き用紙（白紙）を１枚配布します．
   You are given three answer sheets (ruled paper) and one draft sheet (white paper).
5. 問題は全部で５問あります．このうち，３問選択すること．問題ごとに解答用紙を分けて記入すること．
   There are five problems. Select three problems. Write your answer to each problem in a different answer sheet.
6. 解答用紙に解答を記述する際に，問題番号を必ず明記すること．
   When writing the answers, clearly label the problem number on each answer sheet.

平成３０年８月２３日

問題 I の解答に使用する解答用紙の先頭には「問題 I」と明記すること．この解答用紙には問題 I に対する解答以外を記述しないこと．

問題 I　次の行列 A に関して，以下の問いに答えなさい．

$$A = \begin{pmatrix} 2 & -2 & 1 & 1 \\ -2 & 2 & 1 & -3 \\ 0 & 0 & -1 & 1 \\ 1 & -1 & -1 & 2 \end{pmatrix}$$

(1) 行列 A の階数 $\operatorname{rank} A$ を求めなさい．

(2) 行列 X を，階数が $4 - \operatorname{rank} A$ の行列とし，行列 O を零行列とする．このとき，$AX = O$ となる行列 X を一つ求めなさい．

(3) 行列 Y を，行の数が $\operatorname{rank} A$ と等しく，列の数が 4，また階数が $\operatorname{rank} A$ の行列とする．このとき，$YX = O$ となる行列 Y を一つ求めなさい．ただし，行列 O の行と列の数は，設問 (2) の行列 O の行と列の数と等しくなくてもよい．

問題 II の解答に使用する解答用紙の先頭には「問題 II」と明記すること．この解答用紙には問題 II に対する解答以外を記述しないこと．

問題 II

(1) 以下の (a), (b) の定積分を求めなさい．

(a) $\displaystyle\int_0^{\pi} \frac{\sin x}{1+\cos^2 x}\, dx$　　(b) $\displaystyle\int_0^{\pi} \frac{x \sin x}{1+\cos^2 x}\, dx$

(2) 偏微分可能な 2 変数関数 $f(u,v)$ を用いて $f(2x-z, 3y-z) = 0$ と表される xyz 空間における曲面 S について考える．以下の設問 (a), (b) に答えなさい．ただし，解答中で $f(u,v)$ の $u,\ v$ に関する偏微分として各々 $f_u(u,v),\ f_v(u,v)$ を用いなさい．

(a) 曲面 S 上の点 (x_0, y_0, z_0) における法線ベクトルと接平面の式を求めなさい．

(b) 曲面 S の任意の接平面は原点を通るある直線 L に平行であることを示しなさい．また，その直線 L の式を求めなさい．

問題 III の解答に使用する解答用紙の先頭には「問題 III」と明記すること．この解答用紙には問題 III に対する解答以外を記述しないこと．

問題 III　$\mathbb{N}$ を自然数の集合（0 を含む）とし，関数 $\mathrm{ack} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ を以下のように定義する．

$$\mathrm{ack}(m,n) = \begin{cases} n+1 & (m = 0) \\ \mathrm{ack}(m-1,1) & (m > 0 \text{ かつ } n = 0) \\ \mathrm{ack}(m-1,\mathrm{ack}(m,n-1)) & (m > 0 \text{ かつ } n > 0) \end{cases}$$

また，集合 $S = \{x \in \mathbb{N} \mid x < 5\}$ とし，関数 $f : S \to S$ を $f(n) = \mathrm{ack}(1,n) \bmod 5$ と定義する．ここで $a \bmod b$ は自然数 a を自然数 $b \neq 0$ で割った余りを表すものとする．さらに，二項関係 $R \subseteq S \times S$ を $R = \{\langle x, f(x)\rangle \mid x \in S\}$ と定義する．このとき以下の問いに答えなさい．

(1) $\mathrm{ack}(2,0)$ の値を求めなさい．計算過程も示すこと．

(2) すべての $n \in \mathbb{N}$ について，$\mathrm{ack}(1,n) = n+2$ が成立することを証明しなさい．

(3) 関数 f が全単射であることを証明しなさい．また，関数 f の逆関数 g を求めなさい．

(4) 反射律・推移律・対称律をすべて満たす二項関係のことを一般に同値関係という．R が同値関係でないことを反例を 1 つ挙げて示しなさい．

(5) 関係の合成を記号「$\circ$」で表す．関係 $R \circ R \circ R \circ R \circ R$ が同値関係であることを証明しなさい．

問題 IV の解答に使用する解答用紙の先頭には「問題 IV」と明記すること．この解答用紙には問題 IV に対する解答以外を記述しないこと．

問題 IV　二分木を用いて int 型の値の集合を格納するデータ構造を C 言語で実装する．図 1 に，二分木の各ノードの構造体 node の定義を示す．構造体 node は，表 1 に示した関数で操作される．その関数の実装を図 2 に示す．ただし，関数 malloc は常に成功するものとする．このとき以下の問いに答えなさい．

```
struct node {
    int value;
    struct node *left;
    struct node *right;
};
```

図 1: 構造体 node の定義

表 1: 構造体 node を操作する関数

関数	説明

`struct node *new_node(`
`  int value,`
`  struct node *left,`
`  struct node *right`
`)`
	新たなノードを作成し，値 value，左の子ノードへのポインタ left，右の子ノードへのポインタ right を設定し，作成したノードのポインタを返す．

`void traverse(`
`  struct node *n`
`)`
	ポインタ n が示すノードを根とする二分木に含まれるノード群の値を中順にて標準出力に出力する（出力の順序は，左の部分木に格納された値の集合，根の値，右の部分木に格納された値の集合とする）．

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

struct node *new_node(int value, struct node *left, struct node *right)
{
    struct node *n = malloc(sizeof(struct node));
    n->value = value;
    n->left = left;
    n->right = right;
    return (n);
}

void traverse(struct node *n)
{
    if (n == NULL) return;
    traverse(n->left);
    printf("%d\n", n->value);
    traverse(n->right);
}
```

図 2: 表 1 に示した関数の実装

(1) 以下に示す関数 `main()` が実行されたとき，この関数が標準出力に出力する内容を答えなさい．

```
int main()
{
    struct node *top;
    top = new_node(5, new_node(9, NULL, NULL),
                      new_node(7, NULL, NULL));
    traverse(top);
    return (0);
}
```

(2) 二分木を深さ優先で走査する方法としては，関数 `traverse()` で実装した中順以外に，前順と後順がある．前順で走査する関数と後順で走査する関数を以下の表に示す．これらの関数を定義しなさい．

関数	説明

`void pre_order_traverse(`
`  struct node *n`
`)`
	ポインタ `n` が示すノードを根とする二分木に含まれるノード群の値を前順にて標準出力に出力する（出力の順序は，根の値，左の部分木に格納された値の集合，右の部分木に格納された値の集合とする）．

`void post_order_traverse(`
`  struct node *n`
`)`
	ポインタ `n` が示すノードを根とする二分木に含まれるノード群の値を後順にて標準出力に出力する（出力の順序は，左の部分木に格納された値の集合，右の部分木に格納された値の集合，根の値とする）．

(3) 関数 `traverse()` を用いて値が昇順で出力されるような二分木となるように値を挿入する関数 `insert()` を以下の表に示す．ただし，既に二分木に格納されている値と同じ値がこの関数に与えられることはないものとする．関数 `insert()` が正しく動作するように，以下の空欄 (a) ～ (e) を埋めなさい．

関数	説明

`struct node *insert(`
`  struct node *top,`
`  int value`
`)`
	ポインタ top が NULL のときは新しくノードを作成し，そのノードに値を格納し，そのノードへのポインタを返す．それ以外の場合，ポインタ top が示すノードに格納されている値よりも値 value が小さい場合には左の部分木に値 value を格納するために再帰的にこの関数を呼んでポインタ top を返し，大きい場合には右の部分木に値 value を格納するために再帰的にこの関数を呼んでポインタ top を返す．

```
struct node *insert(struct node *top, int value)
{
  if (top == NULL)
    return (new_node(value, NULL, NULL));
  if (value < [ (a) ])
    [ (b) ] = insert([ (c) ]);
  else
    [ (d) ] = insert([ (e) ]);
  return (top);
}
```

(4) 設問 (3) で作成した関数 insert() を用いて空の二分木に 3 つの値 5，7，9 を挿入する関数 main() を以下に示す．この関数を実行したときに，関数 insert() が呼び出される回数（関数 insert() が再帰呼び出しにより呼び出される回数も含める）を答えなさい．

```
int main()
{
    struct node *top = NULL;
    top = insert(top, 5);
    top = insert(top, 7);
    top = insert(top, 9);
    traverse(top);
    return (0);
}
```

(5) 設問 (4) で示した関数 main() では， 空の二分木に 3 つの値 5，7，9 をこの順序で挿入している．この順序を変えることで関数 insert() が呼び出される回数が変化する．3 つの値 5，7，9 を空の二分木に挿入する際に，関数 insert() が呼び出される回数が最小となる順序を 1 つ答えなさい．

(6) 設問 (3) で示した関数 insert() を用いて，空の二分木に関数 insert() が呼び出される回数が最小となる順序で N 個の異なる値を挿入したとする．この二分木に，さらにもう 1 個の値を挿入したときに関数 insert() が呼び出される回数を N を用いて答えなさい．

問題 V の解答に使用する解答用紙の先頭には「問題 V」と明記すること．この解答用紙には問題 V に対する解答以外を記述しないこと．

問題 V　入力値が素数である場合に 1，そうでない場合に 0 を出力する組み合わせ回路を作成する．ただし，入力値が 0 の場合は 0 を出力する．入力値を 4 桁の 2 進数で表したものを最上位桁から $a_3a_2a_1a_0$ とし，出力を p と表すとき，以下の問いに答えなさい．

(1) 以下の真理値表を完成させなさい．

$a_3a_2a_1a_0$	p
0000	0
0001	0
0010	1
1000	0

(2) 設問 (1) の真理値表をもとに，以下のカルノー図を完成させなさい．

a_3a_2 \ a_1a_0	00	01	11	10
00				
01				
11				
10				

(3) a_3，a_2，a_1，a_0 を用いて，p の論理式の例を一つ，加法標準形で示しなさい．ただし，その論理式は 4 つの項の和からなり，各項は 3 つの変数（またはその否定）の積であるものとする．

(4) 設問 (3) で答えた論理式を以下の論理回路記号を用いて表しなさい．

Write the answers to Problem I on one answer sheet, and clearly label it at the top of the page as "Problem I." Do not write answers to other problems on the answer sheet.

Problem I Answer the following questions regarding the matrix A;

$$A = \begin{pmatrix} 2 & -2 & 1 & 1 \\ -2 & 2 & 1 & -3 \\ 0 & 0 & -1 & 1 \\ 1 & -1 & -1 & 2 \end{pmatrix}.$$

(1) Find the rank of the matrix A.

(2) Let X be a matrix whose rank is $4 - \operatorname{rank} A$, and O be a zero matrix. Find a matrix X such that $AX = O$.

(3) Let Y be a matrix where the number of rows equals $\operatorname{rank} A$, the number of columns equals four, and the rank equals $\operatorname{rank} A$. Find a matrix Y such that $YX = O$. The number of columns and rows of the O are not needed to be the same as those of O in Question (2).

Write the answers to Problem II on one answer sheet, and clearly label it at the top of the page as "Problem II." Do not write answers to other problems on the answer sheet.

Problem II

(1) Evaluate the definite integrals (a) and (b) below.

(a) $\displaystyle\int_0^{\pi} \frac{\sin x}{1+\cos^2 x}\, dx$ (b) $\displaystyle\int_0^{\pi} \frac{x \sin x}{1+\cos^2 x}\, dx$

(2) Consider the surface S in the xyz space expressed as $f(2x-z, 3y-z) = 0$ using the partially differentiable function $f(u, v)$ of two variables. Answer Questions (a) and (b) below. In the answers, use the symbols $f_u(u, v)$ and $f_v(u, v)$ to denote the partial derivatives of $f(u, v)$ with respect to u and v, respectively.

(a) Find the normal vector and the equation of the tangent plane to the surface S at the point (x_0, y_0, z_0) on the surface S.

(b) Show that any tangent plane to the surface S is parallel to some line L passing through the origin. Also, find the equation of the line L.

Write the answers to Problem III on one answer sheet, and clearly label it at the top of the page as "Problem III." Do not write answers to other problems on the answer sheet.

Problem III Let $\mathbb{N}$ be the set of natural numbers (including 0) and $\mathrm{ack} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be the function defined as follows:

$$\mathrm{ack}(m, n) = \begin{cases} n + 1 & (m = 0) \\ \mathrm{ack}(m - 1, 1) & (m > 0 \text{ and } n = 0) \\ \mathrm{ack}(m - 1, \mathrm{ack}(m, n - 1)) & (m > 0 \text{ and } n > 0). \end{cases}$$

We also define the set S to be $\{x \in \mathbb{N} \mid x < 5\}$ and the function $f : S \to S$ to be $f(n) = \mathrm{ack}(1, n) \bmod 5$. Here, $a \bmod b$ represents the remainder when a natural number a is divided by a natural number $b \neq 0$. Let the binary relation $R \subseteq S \times S$ be $R = \{\langle x, f(x)\rangle \mid x \in S\}$. Answer the following questions.

(1) Calculate the value of $\mathrm{ack}(2, 0)$. Show the calculation steps.

(2) Prove that $\mathrm{ack}(1, n) = n + 2$ holds for all $n \in \mathbb{N}$.

(3) Prove that the function f is bijective and find the inverse function g of f.

(4) A binary relation is called an equivalence relation if it is reflexive, transitive, and symmetric. Show that R is not an equivalence relation by finding a counterexample.

(5) The symbol "$\circ$" is used to denote the composition of relations. Prove that the relation $R \circ R \circ R \circ R \circ R$ is an equivalence relation.

Write the answers to Problem IV on one answer sheet, and clearly label it at the top of the page as "Problem IV." Do not write answers to other problems on the answer sheet.

Problem IV A data structure that stores `int` values using a binary tree is implemented in the C language. Each node of the binary tree uses the structure `node` defined in Figure 1. The structure `node` is manipulated with the functions in Table 1. Figure 2 shows the implementation of the functions. Note that the function `malloc` always succeeds. Answer the following questions.

```
struct node {
    int value;
    struct node *left;
    struct node *right;
};
```

Figure 1: The definition of the structure `node`.

Table 1: The functions that manipulate the structure `node`.

Function	Description

`struct node *new_node(`
`  int value,`
`  struct node *left,`
`  struct node *right`
`)`
	Create a new node, assign the stored value to `value`, set the pointer of the left child node as `left`, set the pointer of the right child node as `right`, and return the pointer of the created node.

`void traverse(`
`  struct node *n`
`)`
	Print out the values stored in the tree, whose root node is pointed to by `n`, to standard output in in-order (i.e., the order of the output is the set of values stored in the left subtree, the value of the root node, and the set of values stored in the right subtree).

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

struct node *new_node(int value, struct node *left, struct node *right)
{
    struct node *n = malloc(sizeof(struct node));
    n->value = value;
    n->left = left;
    n->right = right;
    return (n);
}

void traverse(struct node *n)
{
    if (n == NULL) return;
    traverse(n->left);
    printf("%d\n", n->value);
    traverse(n->right);
}
```

Figure 2: The implementation of the functions shown in Table 1.

(1) Answer the output on the standard output when the following function `main()` is executed.

```
int main()
{
    struct node *top;
    top = new_node(5, new_node(9, NULL, NULL),
                   new_node(7, NULL, NULL));
    traverse(top);
    return (0);
}
```

(2) Considering methods for depth-first traversal of a binary tree, besides the in-order implemented by the function `traverse()`, there are also pre-order and post-order. The functions which traverse the binary tree in pre-order and post-order are shown in the following table. Define these functions.

Function	Description

`void pre_order_traverse(`
`  struct node *n`
`)`
	Print out the values of the nodes stored in the binary tree, whose root node is pointed to by `n`, to standard output in pre-order (i.e., the order of the output is the value of the root node, the set of values stored in the left subtree, and the set of values stored in the right subtree).

`void post_order_traverse(`
`  struct node *n`
`)`
	Print out the values of the nodes stored in the binary tree, whose root node is pointed to by `n`, to standard output in post-order (i.e., the order of the output is the set of values stored in the left subtree, the set of values stored in the right subtree, and the value of the root node).

(3) The function `insert()` shown in the following table inserts a value to a binary tree so that the output of the function `traverse()` is in ascending order. We assume that any value already stored in the binary tree is not given to the function. Fill in the blanks [(a)] to [(e)] below to complete this function.

Function	Description

`struct node *insert(`
`  struct node *top,`
`  int value`
`)`
	When the pointer `top` is `NULL`, create a node, assign the value `value` to the node, and return the pointer of the node. Otherwise, when the value `value` is less than the value stored in the node pointed by `top`, call this function recursively to store the value `value` in the left subtree and return the pointer `top`; when the value `value` is greater than the value stored in the node pointed by `top`, call this function recursively to store the value `value` in the right subtree and return the pointer `top`.

```
struct node *insert(struct node *top, int value)
{
  if (top == NULL)
    return (new_node(value, NULL, NULL));
  if (value < [   (a)   ])
    [   (b)   ] = insert([   (c)   ]);
  else
    [   (d)   ] = insert([   (e)   ]);
  return (top);
}
```

(4) The function `main()` that inserts three values 5, 7, and 9 in an empty binary tree using the function `insert()` implemented in Question (3) is shown in the following figure. Answer the number of times that the function `insert()` is called when this function is executed (including the number of recursive calls to the function `insert()`).

```
int main()
{
    struct node *top = NULL;
    top = insert(top, 5);
    top = insert(top, 7);
    top = insert(top, 9);
    traverse(top);
    return (0);
}
```

(5) The function `main()` shown in Question (4) inserts three values 5, 7, and 9 in this order to an empty binary tree. The number of calls to the function `insert()` may vary according to the order of insertion. Answer an insertion order of the values 5, 7, and 9 to an empty binary tree that minimizes the number of times that the function `insert()` is called.

(6) Assume that N different values were inserted to an empty binary tree in an order that minimizes the number of calls to the function `insert()` shown in Question (3). Answer, in terms of N, the number of times that the function `insert()` is called when another value is inserted to this binary tree.

Write the answers to Problem V on one answer sheet, and clearly label it at the top of the page as "Problem V." Do not write answers to other problems on the answer sheet.

Problem V Let us develop a combinational logic circuit such that the output becomes 1 if the input is a prime number, and becomes 0 otherwise. Here, when the input is 0, the output becomes 0. The input is expressed as four digits, from the most significant bit, $a_3a_2a_1a_0$ and the output is p. Answer the following questions.

(1) Complete the following truth table.

$a_3a_2a_1a_0$	p
0000	0
0001	0
0010	1
1000	0

(2) Complete the following Karnaugh map corresponding to the truth table in Question (1).

a_3a_2 \ a_1a_0	00	01	11	10
00				
01				
11				
10				

(3) Write an example logical formula of p in the disjunctive normal form using a_3, a_2, a_1, and a_0. Here, this formula is a disjunction of four terms and each term is a conjunction of three variables (or their negation).

(4) Illustrate the formula answered in Question (3) using the following logic circuit symbols.