平成１９（２００７）年度
東京大学大学院学際情報学府学際情報学専攻
修士課程（総合分析情報学コース）
入学試験問題
専　門　科　目

（平成１９年１月２７日　１０：００～１２：００）

試験開始の合図があるまで問題冊子を開いてはいけません。開始の合図があるまで、下記の注意事項をよく読んでください。
(Please read the instructions on the backside.)

１．本冊子は、総合分析情報学コースの受験者のためのものである。
２．本冊子の本文は１６ページである。落丁、乱丁、印刷不鮮明の箇所などがあった場合には申し出ること。
３．本冊子には、計７問の問題が収録されている。この７問の中から４問を選択して解答すること。
４．本冊子の問題には、日本語文と英語文があるが、日本語文が正式なもので、英語文はあくまでも参考である。両者に意味の違いがある場合は、日本語文を優先すること。
５．解答用紙は４枚ある。選択した問題ごとに解答用紙１枚を使用すること。このほかに計算用紙が１枚ある。なお、解答用紙のみが採点の対象となる。
６．解答用紙の上方の欄に、選択した問題の番号及び受験番号を必ず記入すること。問題番号及び受験番号を記入していない答案は無効である。
７．解答には必ず黒色鉛筆（または黒色シャープペンシル）を使用すること。
８．解答は原則として日本語によるものとする。ただし、英語で解答しても採点の対象とする。
９．試験開始後は、中途退場を認めない。
１０．本冊子、解答用紙、計算用紙は持ち帰ってはならない。
１１．次の欄に受験番号と氏名を記入せよ。

受験番号	
氏　　名	

総合分析情報学 第1問 (Question A1)

1枚のコインを投げ，表が出たら1点，裏が出たら0点を得るという試行を n 回繰り返し，得点の合計を X とする。1回の試行で表が出る確率を p とするとき，以下の問いに答えよ。

(1) $n = 10$, $p = 1/2$ のとき，

(a) $X = 5$ となる確率を求めよ。

(b) $X \leq 2$ となる確率を求めよ。

(2) このコイン投げを例に，大数の法則について説明せよ。

(3) このコインが本当に「公正」なものかどうかを調べるため，10回コインを投げ，表の出る回数を数えたところ8回であった。この結果から，表と裏の出る確率に有意な差があるといえるかどうか論ぜよ。有意水準は 5%とし，議論の過程を詳しく述べること。

Question A1

Suppose that you toss a coin and get one point if you get head and zero points if tail. Let X be the total points you get after repeating this trial 10 times, and the probability of getting head on each trial be p.

(1) When $n = 10$ and $p = 1/2$:

(a) What is the probability of $X = 5$?

(b) What is the probability of $X \leq 2$?

(2) Explain *the law of large numbers* using an example of the coin toss.

(3) To examine whether this coin is "fair," you tossed the coin 10 times and got 8 heads. On the basis of this result, can you say that the probabilities of getting heads and tails are significantly different? Test at the .05 level of significance.

総合分析情報学 第 2 問 (Question A2)

（1） A*アルゴリズムの概略を記述せよ。

（2） A*アルゴリズムでは、ある基準が満たされれば、最適な解を見つけることが知られている。その基準とはどのようなものか簡潔に記述せよ。

（3） A*アルゴリズムとダイクストラ法（あるいは均一コスト探索）との違いを簡潔に記述せよ。

Question A2

(1) Describe A* search algorithm.

(2) It is known that A* search will converge into the optimal solution if a certain condition is satisfied. Concisely describe what this condition is.

(3) Describe the difference between A* search and Dijkstra method (or Uniform-cost search).

総合分析情報学 第3問 (Question A3)

コンピュータの処理の並列化について以下の問いに答えよ。

（1） 命令レベルの並列化の代表的な手法を説明せよ。

（2）プロセッサレベルの並列化にマルチプロセッサがある。マルチプロセッサの代表的な構成方法として、共有メモリ型と分散メモリ型がある。

(a) 共有メモリ型マルチプロセッサと、分散メモリ型マルチプロセッサの構成方法がどのようなものか説明せよ。

(b) これらを、ハードウェアやソフトウェアの複雑さ、プログラミングの容易さや安全性の観点から比較して、利害得失を述べよ。

（3）マルチプロセッサの構成方法には、対称型マルチプロセッサと非対称型マルチプロセッサがあるが、それぞれの方式に適した典型的な適応例を挙げ、その応用に即してそれぞれがどのような方式かを説明せよ。

Question A3

Answer the following problems on parallel processing in computers.

(1) Explain representative methods for instruction-level parallelism.

(2) Multi-processor architecture is the most popular approach for processor-level parallelism. Multi-processor architectures are classified into shared-memory architecture and distributed-memory architecture.
    (a) Explain shared-memory architecture and distributed-memory architecture.
    (b) Explain advantage and disadvantage of these architectures by comparing both architectures from point of views of complexity of hardware and software, easiness of programming and software security.

(3) Multi-processor systems are classified into symmetric multi-processor system and asymmetric multi-processor system. Explain architecture of each system, describing typical example applications of each system.

総合分析情報学 第4問 (Question A4)

N個のバッファに対して、2つのタスク *producer* と *consumer* が次の動作を並行に行う。*producer* は、データを1つずつ生成しバッファに格納し続ける。*consumer* はデータをバッファから1つずつ取り出して消費し続ける。バッファのデータの個数がゼロになると *consumer* は動作できない。逆に、データの個数が *N* 個たまると *producer* は動作できない。そこで、*producer*、*consumer* 間で同期をとる必要がある。これをproducer-consumer 問題という。

（1）以下のサンプルコードはproducer-consumer 問題を解こうとしているが、うまく動かないケースがある。どのようなケースか説明せよ。以下で使われているsleep()関数は、それを呼び出したタスクの動作を停止させ、wakeup(task)は引数で指定したタスクの動作が停止していたら再開させる。

```
#define NBUF 100
int num = 0;
```

```
void producer()
{
 int item;

 loop:
   produceItem(&item);
   if(num == NBUF) sleep();
    putItem(item);
    num = num + 1;
    if(num==1)wakeup(consumer);
   goto loop;
}
```

```
void consumer()
{
 int item;

 loop:
   if(num == 0) sleep();
    getItem(&item);
    num = num - 1;
    if(num==NBUF -1)
       wakeup(producer);
    consumeItem(item);
   goto loop;
}
```

（2） 以下はビジーウェイティングによる解決策であるが、本方法の問題点を述べよ。

```
#define NBUF 100
int num = 0;
```

```
void producer()
{
 int item;

 loop:
   produceItem(&item);
   while(num == NBUF);
    /*do nothing*/
    putItem(item);
    num = num + 1;
   goto loop;
}
```

```
void consumer()
{
 int item;

 loop:
   while(num ==0);
    /*do nothing*/
    getItem(&item);
    num = num - 1;
    consumeItem(item);;
   goto loop;
}
```

（3）次に示す P 命令と V 命令を持つ計数型セマフォを 3 つ利用して、producer-consumer 問題の解を記せ。

- P (counter) 命令：counter が 0 より大きければ、counter を 1 減らす。counter が 0 以下ならば、現在のタスクをキューに入れ、他のタスクの実行に切り替える。この処理はアトミックに実行される。
- V (counter) 命令：タスクキューが空ならば、counter を 1 増やす。空でなければ、タスクキューからタスクを一つ取りだし、そのタスクの実行に切り替える。この処理はアトミックに実行される。

Question A4

Two concurrent tasks "producer" and "consumer" make the following operations against an N-item buffer independently. The producer task generates a data item, and puts it into the buffer repeatedly. The consumer task gets a data item from the buffer, and consumes it repeatedly. If the number of stored items in the buffer becomes zero, the consumer task stops. On the other hand, if the buffer contains N items in it, the producer task cannot proceed any more. Therefore, the producer and consumer must synchronize with each other. This is a so-called *producer-consumer problem.*

(1) The following program attempts to solve the producer-consumer problem. Point out any problems in this program. Note, in the following code, two system calls are used to achieve synchronization; `sleep()` suspends the caller task, and `wakeup(task)` restarts the specified task if the task has been suspended.

```
#define NBUF 100
int num = 0;
```

```
void producer()
{
 int item;

 loop:
   produceItem(&item);
   if(num == NBUF) sleep();
    putItem(item);
    num = num + 1;
   if(num==1)wakeup(consumer);
   goto loop;
}
```

```
void consumer()
{
 int item;

 loop:
   if(num == 0) sleep();
    getItem(&item);
    num = num - 1;
    if(num==NBUF -1)
       wakeup(producer);
    consumeItem(item);
   goto loop;
}
```

(2) The following code attempts to achieve the producer-consumer problem using busy-wait. Point out any problems in this program.

```
#define NBUF 100
int num = 0;
```

```
void producer()
{
  int item;

   loop:
    produceItem(&item);
    while(num == NBUF);
    /*do nothing*/
    putItem(item);
    num = num + 1;
  goto loop;
}
```

```
Void consumer()
{
  int item;

  loop:
    while(num ==0);
     /*do nothing*/
    getItem(&item);
    num = num - 1;
    consumeItem(item);;
  goto loop;
}
```

(3) Write a program to solve the producer-consumer problem using three semaphores of the following P and V functions.

- `P (counter)`: If `counter` is greater than `0`, this function increments the counter. If `counter` is less than zero, the caller task puts itself into a waiting queue and switches to another task. This function is an atomic operation.
- `V (counter)`: If the task queue is empty, this function increments `counter`; otherwise, it picks up a task from the waiting queue and wakes it up. This function is also an atomic operation.

総合分析情報学 第5問 (Question A5)

（1）抽象データ型（Abstract Data Type）とは何かを説明し、そのプログラミング上の利点を述べよ。

（2）次のインタフェース関数を備える複素数を表す抽象データ型を定義せよ。

抽象データ型名： complex
加算関数： complexAdd(a, b)
関数の戻り値： complex 型
引数（a, b）： complex 型
乗算関数： complexMult(a, b)
関数の戻り値： complex 型
引数（a, b）： complex 型

（3）抽象データ型を発展させたオブジェクト指向型言語の重要な特徴に継承がある。継承とはどのような機構か説明せよ。

（4）多重継承と単一継承の違いを説明し、その利害得失を論ぜよ。

Question A5

(1) Define *abstract data type*, and describe its advantage from a programming point of view.

(2) Define an abstract data type for complex number system using the following interfaces.

Abstract data type name : `complex`
Add function : `complexAdd(a, b)`
Type of return value : `complex`
Types of parameters (a, b) : `complex`
Multiply function : `complexMult(a, b)`
Type of return value : `complex`
Types of parameters (a, b) : `complex`

(3) Object-oriented languages, which are derived from abstract data type mechanism, are equipped with inheritance mechanism. Explain inheritance mechanism.

(4) Explain the difference between single-inheritance and multiple-inheritance, and describe advantage and disadvantage of each mechanism.

総合分析情報学 第6問 (Question A6)

インターネットにおけるネットワークプロトコルについて以下の問いに答えよ。

（1）プロトコルスタックはレイヤー構造になっているが、この構造の長所と短所を簡潔に記せ。
（2）各レイヤーの呼称とその特徴について簡潔に述べよ。
（3）「砂時計モデル」とは何か？
（4）パケットスイッチングとサーキットスイッチングを両者の違いがわかるように説明せよ。インターネットではどちらが主に使用されているか理由を含めて述べよ。
（5）パケットスイッチングにおいて、パケットロスが生じる原因を説明せよ。
（6）パケットスイッチングにおいて、パケット遅延が生じる原因を説明せよ。

Question A6

Answer the following questions regarding the Internet network protocols.

(1) The Internet protocol stack forms a layered structure. Briefly explain advantages and disadvantages of the layered structure.
(2) Name each layer in the Internet protocol stack and briefly explain the characteristics of each layer.
(3) Define "the waist of the hourglass" model.
(4) Explain packet switching and circuit switching, clarifying the difference between these two. Answer which switching technique is being used in the current Internet and justify the reason for that.
(5) In packet switching, why and how packet loss occurs?
(6) In packet switching, why and how packet delay occurs?

総合分析情報学 第７問 (Question A7)

表１と図１は、３系統の入力 A、B、Carry in と２系統の出力 S と Carry out を持つ、１ビット全加算器の真理値表とゲート回路図である。

A	B	Carry in	S	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

表１：１ビット全加算器の真理値表

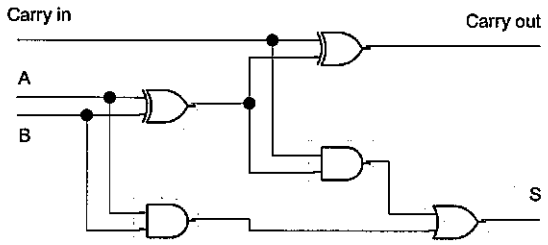

図１：１ビット全加算器のゲート構成図

（１）１ビットの全加算器を組み合わせて４ビットの全加算器を構成せよ。

（２） 単に全加算器を組み合わせて８、１６、３２ビットの全加算器を構成した場合、ビット数が増えるにつれて加算器の遅延がどのようになるか述べよ。

（３）キャリー先読み（Carry Look Ahead）回路は、全加算器の計算の遅延を減少させることができる。そこで、４ビットの全加算器のキャリー先読み回路の真理値表を作成せよ。

（４）４ビットの全加算器のキャリー先読み回路を設計せよ。

（５）キャリー先読み回路を用いると、真理値表だけから考えると、ビット数 N の全加算器が、N によらず一定（定数）時間で実現できそうだが、実際はそうはいかないのはなぜか説明せよ。

Question A7

Table 1 and Figure 1 show the truth table and circuit diagram of 1-bit full-adder with three inputs A, B, and carry-in and two outputs S and carry out.

A	B	Carry in	S	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 1 : Truth table of 1-bit full-adder

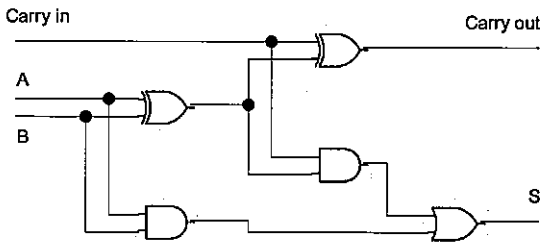

Figure 1 : 1-bit Full-adder circuit

(1) Design four bits full-adder with multiple 1-bit full-adders.

(2) Explain the computation delay caused according to the increase in bit numbers when we design full-adders of 8, 16, and 32 bits by simply with multiple 1-bit full-adders.

(3) *Carry look-ahead circuit* can decrease the delay in multiple-bit full-adders. Describe truth table of carry look-ahead circuit of four-bit full-adders.

(4) Design a carry look-ahead circuit of four-bit full-adders.

(5) By considering only truth tables, it may appear that computation time of N-bits full-adder is static independent of N, but it is not the case. Explain why the computation time cannot be static.

Entrance Examination for Masters Program in Applied Computer Science Course, Graduate School of Interdisciplinary Information Studies, The University of Tokyo. Academic Year 2007 (10:00-12:00, January 27th, 2007)

Directions: Do not open this booklet before the examination begins.
Read the following instructions carefully.

1. This booklet is for the examinees in Applied Computer Science Course, Graduate School of Interdisciplinary Information Studies.
2. This booklet includes sixteen pages. Report missing, misplaced, and imperfect pages to the instructor.
3. This booklet includes seven questions. Select any four questions and answer only those four.
4. Each question is described both in Japanese and in English. Use the Japanese version primarily; the English version is provided for the reference purpose only.
5. There are four answer sheets and a scratch paper. Use one answer sheet per question. A scratch paper is provided for calculation. Only the answer sheets will be considered valid.
6. Write a question number and your examinee's number in the designated boxes located at the top of each answer sheet. The answer missing a question number and/or an examinee's number will not be considered valid.
7. Use only black pencils (or black mechanical pencils).
8. Answer the questions in Japanese as a general rule, although you are also allowed to answer in English.
9. Do not leave the room until the examination is finished.
10. Do not take away this booklet, the answer sheets, and the scratch paper.
11. Write your examinee's number and your name in the designated boxes below.

Examinee's Number	
Name	