理工学研究科　博士課程前期課程

2026 年 4 月・2025 年 9 月入学　大学院夏季入学試験問題
理工学研究科　前期課程　情報工学専攻
英　　語

（注）問題番号または記号を必ず解答用紙に明記すること

問題 I　次の英文を読み、英文の内容に従って問いに答えよ。

Let's detail the derivation of the 2D vector (x_1, y_1) obtained by rotating an initial 2D vector (x_0, y_0) counter-clockwise around the origin by an angle of θ radians.

[Part 1]

First, any 2D vector (x_0, y_0) can be expressed as the sum of a vector along the x-axis, $(x_0, 0)$, and a vector along the y-axis, $(0, y_0)$, i.e., $(x_0, y_0) = (x_0, 0) + (0, y_0)$. Leveraging this property, the rotated vector (x_1, y_1) can be found by adding the vector obtained from rotating $(x_0, 0)$ by an angle θ and the vector obtained from rotating $(0, y_0)$ by an angle θ. Next, let's consider rotating the vector $(x_0, 0)$, which lies on the x-axis, by an angle θ. When a point at a distance of x_0 from the origin rotates by an angle θ, its new x-coordinate becomes $x_0 \cos\theta$ and its new y-coordinate becomes $x_0 \sin\theta$. Therefore, the vector obtained by rotating $(x_0, 0)$ by an angle θ is $(x_0 \cos\theta, x_0 \sin\theta)$.

Similarly, let's consider rotating the vector $(0, y_0)$, which lies on the y-axis, by an angle θ. When a point at a distance of y_0 from the origin rotates by an angle θ, its new x-coordinate becomes $-y_0 \sin\theta$ and its new y-coordinate becomes $y_0 \cos\theta$. This is because applying the standard rotation matrix to a vector with an x-component of 0 and a y-component of y_0 yields these results. Thus, the vector obtained by rotating $(0, y_0)$ by an angle θ is $(-y_0 \sin\theta, y_0 \cos\theta)$.

By summing the results from the previous steps, we can derive (x_1, y_1):

$$(x_1, y_1) = (x_0 \cos\theta, x_0 \sin\theta) + (-y_0 \sin\theta, y_0 \cos\theta).$$

Summing the respective components, we get:

$$x_1 = x_0 \cos\theta - y_0 \sin\theta, \text{and} \quad y_1 = x_0 \sin\theta + y_0 \cos\theta.$$

Therefore, the 2D vector (x_1, y_1) obtained by rotating an initial 2D vector (x_0, y_0) counter-clockwise around the origin by an angle of θ radians is given by:

$$(x_1, y_1) = (x_0 \cos\theta - y_0 \sin\theta, x_0 \sin\theta + y_0 \cos\theta), \text{ or } \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \boldsymbol{A} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \text{ where } \boldsymbol{A} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

[Part 2]

In a 2D plane, any point (x, y) can be uniquely represented as a complex number $z = x + iy$, where i is the imaginary unit ($i^2 = -1$). Similarly, our initial vector (x_0, y_0) can be written as the complex number $z_0 = x_0 + iy_0$. The rotated vector (x_1, y_1) can be represented as $z_1 = x_1 + iy_1$.

The key to understanding rotation in the complex plane lies with Euler's formula, which states that for any real number θ:

$$e^{i\theta} = \cos\theta + i \sin\theta \ .$$

This complex exponential $e^{i\theta}$ acts as a rotation operator. When you multiply a complex number by $e^{i\theta}$, you effectively rotate that complex number (and thus the corresponding vector) counter-clockwise around the origin by an angle θ.

To rotate the initial complex number $z_0 = x_0 + iy_0$ by an angle θ, we simply multiply it by $e^{i\theta}$:

$$z_1 = z_0 e^{i\theta} \ .$$

Now, let's substitute the complex number representations and Euler's formula:

$$z_1 = (x_0 + iy_0)(\cos\theta + i \sin\theta)$$

Next, we expand this product:

$$\begin{aligned} z_1 &= x_0 \cos\theta + x_0 (i \sin\theta) + i\, y_0 \cos\theta + iy_0\, i \sin\theta \\ &= x_0 \cos\theta + ix_0 \sin\theta + i\, y_0 \cos\theta + i^2 y_0 \sin\theta. \end{aligned}$$

Since $i^2 = -1$, the equation becomes:

$$z_1 = x_0 \cos\theta + ix_0 \sin\theta + i\, y_0 \cos\theta - y_0 \sin\theta \ .$$

Now, we group the real and imaginary parts:

$$z_1 = (x_0 \cos\theta - y_0 \sin\theta) + i(x_0 \sin\theta + y_0 \cos\theta) \ .$$

Recall that $z_1 = x_1 + iy_1$. By comparing the real and imaginary parts of the equation above, we can directly find x_1 and y_1:

$$x_1 = x_0 \cos\theta - y_0 \sin\theta, \text{ and } \quad y_1 = x_0 \sin\theta + y_0 \cos\theta.$$

These are precisely the same rotation formulas we derived earlier using vector decomposition in the Cartesian coordinate system. This demonstrates how representing 2D vectors as complex numbers and using multiplication by $e^{i\theta}$ provides an elegant and concise way to perform rotations.

1. Part 1及びPart 2の説明によく適合する図をそれぞれ1枚描け。図には説明で用いている軸名、座標、角度を必ず記すこと。
2. 説明に基づいて行列$\boldsymbol{A}$の逆行列を導出する説明文を日本語、英語の2通りで作成せよ。但し、説明文の内容は日本語、英語で同一にすること。

（注）問題番号または記号を必ず解答用紙に明記すること

問題 II　次の英文は，文字コードの標準規格である Unicode 標準の規格書 *"The Unicode Standard, Version 15"* から一部を抜粋し，改変したものである．この英文を読み，以下の設問に答えよ．

While taking the ASCII character set as its starting point, the Unicode Standard goes far beyond ASCII's limited ability to encode only the upper- and lowercase letters A through Z. It provides the capacity to encode all characters used for the written languages of the world — more than 1 million characters can be encoded. No escape sequence or control code is required to specify any character in any language. The Unicode character encoding treats alphabetic characters, ideographic characters, and symbols equivalently, which means they can be used in any mixture and with equal facility.

The Unicode Standard specifies a numeric value (code point) and a name for each of its characters. In this respect, it is similar to other character encoding standards from ASCII onward. In addition to character codes and names, other information is crucial to ensure legible text: a character's case, directionality, and alphabetic properties must be well defined. The Unicode Standard defines these and other semantic values, and it includes application data such as case mapping tables and character property tables as part of the Unicode Character Database. Character properties define a character's identity and behavior; they ensure consistency in the processing and interchange of Unicode data.

The Unicode Standard contains 149,186 characters from the world's scripts. These characters are more than sufficient not only for modern communication for the world's languages, but also to represent the classical forms of many languages. The standard includes the European alphabetic scripts, Middle Eastern right-to-left scripts, and scripts of Asia and Africa. Many archaic and historic scripts are encoded. In addition, the Unicode Standard contains many important symbol sets, including currency symbols, punctuation marks, mathematical symbols, technical symbols, geometric shapes, dingbats, and emoji.

The Unicode Standard began with a simple goal: to unify the many hundreds of conflicting ways to encode characters, replacing them with a single, universal standard. The pre-existing legacy character encodings were both inconsistent and incomplete — two encodings could use the same codes for two different characters and use different codes for the same characters, while none of the encodings handled any more than a small fraction of the world's languages. Whenever textual data was converted between different programs or platforms, there was a substantial risk of corruption. Programs often were written only to support particular encodings, making development of international versions expensive.

出典：The Unicode Consortium, *The Unicode Standard, Version 15*, 2022.（第 1 章より抜粋，一部改変．）

補足 1　Unicode 標準は単に Unicode と呼ばれることも多い．
補足 2　ASCII は 1960 年代にアメリカで策定された文字コードである．
補足 3　code point：コードポイント（または符号位置），Basic Multilingual Plane：基本多言語面，dingbat：装飾記号，emoji：絵文字．

1. 第一段落の下線部「It」，「they」が指しているものを，それぞれ英文から抜き出せ．
2. Unicode 標準と ASCII の相違点を第一段落から読み取って日本語で説明せよ．また，Unicode 標準と ASCII の類似点を第二段落から読み取って日本語で説明せよ．
3. 第三段落を日本語に訳せ．
4. 最終段落の内容に則して，下記の問に答えよ．
   (1) Unicode 標準よりも古くから存在していた文字コードはどのような欠点を持っていたか，日本語で説明せよ．
   (2) その欠点はどのような問題を引き起こしていたか，日本語で説明せよ．

2026年4月・2025年9月入学

大学院夏季一般入学試験問題

理工学研究科　前期課程　情報工学専攻

問題Ⅰは必答。
問題Ⅱ～Ⅴのうち、2問を選んで解答すること。

問題番号を解答用紙へ記し、1枚の解答用紙に1問だけ解答すること。

上記事項が守られていない場合、得点が無効となることがありますので、注意してください。

（注）問題番号または記号を必ず解答用紙に明記すること

問題 I（全員解答すること）

1. 3 次正方行列 $A = \begin{pmatrix} 4 & 2 & 2 \\ 2 & 3 & 0 \\ 2 & 0 & 5 \end{pmatrix}$ について以下の問いに答えよ．

(1) 行列 A の固有値を求めよ．

(2) 直交行列 P を用いて行列 A を対角化せよ．行列 P も答えること．

(3) $B^2 = A$ となる行列 B を求めよ．

2. $\mathbb{R}$ を実数全体の集合とし，$f : \mathbb{R} \to \mathbb{R}$ を連続関数とする．任意の $x, y \in \mathbb{R}$ に対して

$$f(x) + f(y) = f(x + y)$$

が成り立つとき，以下の問いに答えよ．ただし $f(1) = c$ である．解答には導出過程も記述すること．

(1) $f(0)$ を求めよ．

(2) 自然数 n について，$f(n)$ を求めよ．

(3) 自然数 m, n について，$f\left(\frac{m}{n}\right)$ を求めよ．

(4) 有理数 r について，$f(r)$ を求めよ．

(5) 実数 x について，$f(x)$ を求めよ．ただし，任意の実数 x について，$n = 1, 2, \ldots$ に対して

$$x - \frac{1}{n} < r_n < x + \frac{1}{n}$$

をみたす有理数 r_n が存在することを利用してよい．

3. 未知関数 $y(x)$ に関する微分方程式

$$\frac{\mathrm{d}y}{\mathrm{d}x} = ay - by^2$$

について以下の問いに答えよ．ただし a, b は正の定数とする．

(1) $u(x) = y^{-1}(x)$ とおく．この変数変換により得られる微分方程式を記述せよ．

(2) (1) の微分方程式を $u(x)$ について解き，$\frac{\mathrm{d}y}{\mathrm{d}x} = ay - by^2$ の一般解を求めよ．

問題 II

下のアルゴリズム 1 は, 2 つの単調非減少な数列から別の数列を作成し出力する.

アルゴリズム 1
入力. 単調非減少な数列 $a = (a_1, a_2, \ldots, a_n)$ と $b = (b_1, b_2, \ldots, b_m)$
出力. 数列 c
1. i と j をどちらも 1 とし, c を空の数列とする.
2. $a_i \leq b_j$ ならば c の末尾に a_i を加え, i の値を 1 増やす. $a_i > b_j$ ならば c の末尾に b_j を加え, j の値を 1 増やす.
3. $i > n$ ならば, c の末尾に $b_j, \ldots, b_m$ を順番通りに加えた後に, c を出力して終了. $j > m$ ならば, c の末尾に $a_i, \ldots, a_n$ を順番通りに加えた後に, c を出力して終了. それ以外ならば, ステップ 2 に戻る.

1. $a = (3, 5, 7, 10, 15)$, $b = (2, 7, 11, 18, 21)$ に対してアルゴリズム 1 を実行したときに出力される数列と, ステップ 2 が何回実行されるか答えなさい.

2. 入力の数列の長さが n と m の場合のアルゴリズム 1 の計算量を, n と m のオーダーで表しなさい.

次のアルゴリズム 2 は, 数列 x を並び替えて単調非減少な数列 y を作成する. ただし, x の長さは 2 の累乗である（つまり, 非負整数 d を用いて 2^d と表される）と仮定している.

アルゴリズム 2
入力. 数列 $x = (x_1, x_2, \ldots, x_k)$（$k$ は 2 の累乗）
出力. 単調非減少な数列 $y = (y_1, y_2, \ldots, y_k)$
1. $k = 1$ ならば数列 x をそのまま数列 y として出力して終了する. $k \geq 2$ ならば, 以下のステップ 2 以降を実行する.
2. a を数列 $(x_1, \ldots, x_{k/2})$, b を数列 $(x_{(k/2)+1}, \ldots, x_k)$ とする.
3. 数列 a を入力としてアルゴリズム 2 を再帰呼び出し, 出力された数列 a' を得る.
4. 数列 b を入力としてアルゴリズム 2 を再帰呼び出し, 出力された数列 b' を得る.
5. 数列 a' と数列 b' を入力としてアルゴリズム 1 を実行し, 出力された数列 y を得る. 数列 y を出力して終了.

例えば数列 $x = (2, 1)$ を入力としてアルゴリズム 2 を実行すると全部で 2 回の再帰呼び出しが行われる. 最初の呼び出しでは数 2 のみからなる数列 (2), 2 回目の呼び出しでは数 1 のみからなる数列 (1) が入力される.

3. 数列 $x = (3, 1, 4, 2)$ を入力としてアルゴリズム 2 を実行した際には, 全部で何回再帰呼び出しが行われるか答えなさい. また, それぞれの再帰呼び出しでの入力はどのような数列であるか答えなさい. 解答の際は, どの数列が何回目の再帰呼び出しの入力か分かるように答えること.

4. 長さ k に対する数列に対するアルゴリズム 2 の計算量 $f(k)$ が $k \geq 2$ のときに満たす漸化式を与えなさい. アルゴリズムの細部について必要に応じて仮定を置いてもいいが, その場合はそれについて説明すること.

5. 小問 4 で与えた漸化式を元に, アルゴリズム 1 の計算量 $f(k)$ を k のオーダーで表しなさい. 結論だけではなく, 証明も与えること.

問題Ⅲ

ネットワーク管理者のA氏は，自身が管理しているネットワーク機器をリレーショナルデータベースで管理している．以下はそのリレーションを示している．

R（機種名，型式番号，製品番号，種類，メーカー，サポートセンター連絡先，導入日，設置場所，IPv4アドレス）

※下線が引かれた属性は主キーを表しており，導入日を除くそれぞれの属性は文字列型，導入日は日付型である．

このリレーションには，「主キー → 非キー属性」の関数従属以外に，

- 機種名，型式番号，製品番号 → 導入日，設置場所
- 機種名，型式番号 → 種類，メーカー
- メーカー → サポートセンター連絡先

という関数従属が存在している．

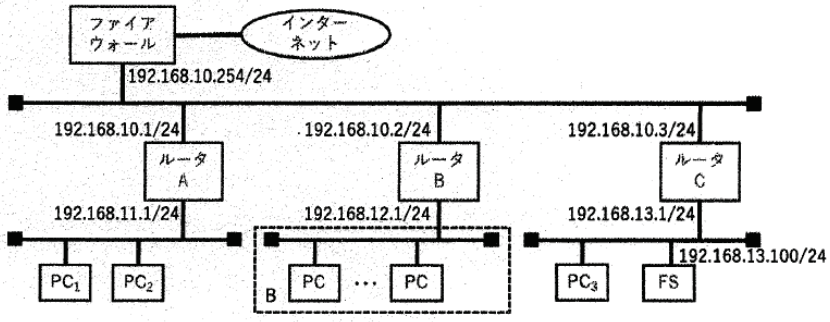
このリレーションでは，購入したばかりで未設置かつ未設定の機器を管理することができないという問題を抱えている．

1．下線部の問題が生じている理由を説明せよ．

2．リレーションRを第3正規形にせよ．それぞれのリレーションにおける主キーを下線で示すこと．なお，リレーションの名称はR1，R2，・・・のように付けること．

3．第3正規形にしたリレーションに対して，192.68.10.0/24のネットワークに接続されている機器の「機種名，型式番号，製品番号，IPv4アドレス，設置場所」を取得するためのSQLを記述せよ．

小問3のSQLにより取得した機器とそれに関連する機器が右図のように接続されていたとする．



4．破線で囲まれたセグメントBに接続可能なPC（パソコン）の台数を答えよ．ただし，ルータBを除きPC以外に接続されている機器はないものとする．

ある日にFS（ファイルサーバ）をリプレースしたところ，PC_1とPC_2からFSにアクセスできないという事象が発生した．それぞれのPCからインターネットには正常にアクセスできている．調査のためにPC_3からFSにアクセスしたところ，こちらは正常にアクセスできた．さらに，PC_1からFSに対してtracerouteコマンドを利用してアクセス確認を行った．その実行結果を右に示す．

```
$ traceroute 192.168.13.100
192.168.13.100 へのルートをトレースしています
経由するホップ数は最大 30 です：
1  2 ms    2 ms    2 ms    192.168.11.1
2  3 ms    3 ms    3 ms    192.168.10.3
3  *    *    *
…
```

これらの調査結果をもとに，PC_1とPC_2からFSにアクセスできない原因となっていた設定の誤りを特定した．その誤りを訂正したところ，PC_1とPC_2から正常にFSにアクセスできるようになった．

5．FSへのアクセス不具合について，原因となっていた設定の誤りとして想定されるものを二つ挙げよ．

（注）問題番号または記号を必ず解答用紙に明記すること

問題 IV

つぎの設問に答えよ．

1. ユークリッドの互除法により 2 個の自然数 x, y の最大公約数 $d = \gcd(x, y)$ を計算する以下の再帰関数（C 言語のプログラム）について考える．

```
1:  long long int gcd(long long int x, long long int y){
2:      if(y==0) return x;
3:      long long int q=x/y;
4:      return gcd(y, x-q*y);
5:  }
```

(1) この関数を「`gcd(345,234);`」と呼び出したときの戻り値を示せ．

(2) この関数を，自然数ではなく負の数を指定して「`gcd(-345,234);`」と呼び出したときの戻り値を示せ．

(3) 入力として自然数ではなく負の整数が与えられても結果が常に非負の数になるように，プログラムを修正せよ．解答には修正すべき行番号を示し，それをどのように修正すればよいのかを記せ．

2. 上記と同様に，$d = \gcd(x, y) = \alpha \cdot x + \beta \cdot y$ を満たす α, β，および d を計算する以下の再帰関数（C 言語のプログラム）について考える．

(1) 空欄 [(i)] および [(ii)] に適切な「式」を記入せよ．

(2) `gcd(x,y)` の値が非負になるようにプログラムを修正せよ．解答には修正すべき行番号を示し，それをどのように修正すればよいのかを記せ．

```
1:  long long int gcd(long long int x,long long int y,
2:                    long long int*alpha,long long int*beta){
3:      long long int q,a,b;
4:      if(y==0) { *alpha=1; *beta=0; return x;}
5:      long long int d=gcd(y, x-(q=x/y)*y, &a, &b);
6:      *alpha=[ (i) ]; *beta=[ (ii) ]; return d;
7:  }
```

3. 以下で指示する値を表示する C 言語のプログラムを記述せよ．

相異なる 3 個の素数 p_0, p_1, p_2 について，p_0 による剰余が r_0，p_1 による剰余が r_1，p_2 による剰余が r_2 であるような整数 v の値を計算する関数「`long long int rem3(long long int p[3], long long int r[3]);`」を C 言語の関数として定義せよ．実引数は `long long int` 型要素 3 個からなる配列 `p[3]` および `r[3]` であり，`p[`i`]` に p_i，`r[`j`]` に r_j $(0 \le i, j \le 2)$ を格納して呼び出すこととする．

ただし，関数定義の中で上記 **2.**(2) で修正後の関数
「`long long int gcd(long long int x, long long int y, long long int*alpha, long long int*beta);`」
を適切に呼び出すこと．なお，入力される数値や途中結果は正しく表現可能な範囲に入っていると仮定してよい．

（注）問題番号または記号を必ず解答用紙に明記すること

問題 V

多値の論理値をとる何らかの原子論理式（atomic formulas）の集合および否定（negation）¬，論理積（conjunction）∧，論理和（disjunction）∨ の三演算から成るブール論理系において，あるものが白馬（white horse）であるときそれは馬であることを，構文論的に（syntactically），すなわち根拠を添えつつ論理式を逐次に書き換える様式にて示せ．その際に，否定と選言から定義される含意（implication）→ および論理的含意（logical implication）⇒ を用いてもよい．

問題 I

1. 次の問いに答えよ.

(1) 次の行列 A の固有値と対応する固有ベクトルを求めよ.

$$A = \begin{pmatrix} 1 & 0 & -2 \\ 53 & 2 & -34 \\ 6 & 0 & -6 \end{pmatrix}$$

(2) (1) の行列 A を対角化した $\widehat{A}$ を, $P^{-1}AP = \widehat{A}$ として求めたい. P と $\widehat{A}$ を答えよ.

(3) 次の実変数実数値関数 $y_1(x)$, $y_2(x)$, $y_3(x)$ に関する連立微分方程式の一般解を求めよ.

$$\begin{aligned} y_1'(x) &= y_1(x) - 2y_3(x) \\ y_2'(x) &= 53y_1(x) + 2y_2(x) - 34y_3(x) \\ y_3'(x) &= 6y_1(x) - 6y_3(x) \end{aligned}$$

2. Fibonacci 数列とは, 次のようにして, 漸化式を用いて定義された数列である.

$F_1 = 1, \quad F_2 = 1, \quad F_k = F_{k-1} + F_{k-2}$ $(k \in \mathbf{Z}, k \geq 3)$

Fibonacci 数列の一般項は次のように表されることが知られている. この式は Binet の公式と呼ばれている.

$$F_k = \frac{1}{\sqrt{5}} \left\{ \left(\frac{1+\sqrt{5}}{2} \right)^k - \left(\frac{1-\sqrt{5}}{2} \right)^k \right\}$$

(1) Binet の公式が $k = 1$ のとき成り立つことを示せ.
(2) Binet の公式が $k = 2$ のとき成り立つことを示せ.
(3) Binet の公式が $k \geq 3$ のときにも成り立つことを数学的帰納法を用いて示せ.

（注）問題番号または記号を必ず解答用紙に明記すること

問題II

以下のプログラムSortは配列に格納された整数値のデータを再帰的に分割し，分割したデータの値の大小を比較しながら併合していくことでデータを昇順に整列するプログラムである．このプログラムは併合に副プログラムMergeを使用する．なお，それぞれのプログラムは擬似コードで記述されている．

```
○Sort(整数型の配列：list, 整数型：num)
  整数型の配列：slist1, slist2
  整数型：i, num1, num2
  if (  ①  )
    num1 ← num ÷ 2
    num2 ←  ②
    i ← 0
    while (i < num1)
      slist1[i] ← list[i]
      i ← i + 1
    endwhile
    i ← 0
    while (i < num2)
      slist2[i] ←  ③
      i ← i + 1
    endwhile
    Sort(slist1, num1)
    Sort(slist2, num2)
    Merge(slist1, num1, slist2, num2, list)
      list の出力処理
  endif
```

```
○Merge(整数型の配列 slist1, 整数型：num1,
       整数型の配列 slist2, 整数型：num2,
       整数型の配列：list)
  整数型：i, j
  i ← 0
  j ← 0
  while (  ④  )
    if (slist1[i] < slist2[j])
      list[i + j] ← slist1[i]
      i ← i + 1
    else
      list[i + j] ← slist2[j]
      j ← j + 1
    endif
  endwhile
  while (i < num1)
    list[  ⑤  ] ← slist1[i]
    i ← i + 1
  endwhile
  while (j < num2)
    list[  ⑥  ] ← slist2[j]
    j ← j + 1
  endwhile
```

＜擬似コードの記述ルール＞

記述形式	説明
○手続き名または関数名	手続きまたは関数を宣言する．
型名：変数名	変数を宣言する．
手続き名または関数名(引数,...)	手続きまたは関数を呼び出し，引数を受け渡す．

if (条件式)
　処理1
else
　処理2
endif
	選択処理を示す．条件式が真であるとき，処理1を実行し，条件式が偽であるとき，処理2を実行する．

while (条件式)
　処理
endwhile
	前判定繰り返し処理を示す．条件式が真である間，処理を繰り返し実行する．

配列	配列の要素には，“[”と“]”の間にアクセス対象要素の要素番号を指定することでアクセスする．

＜擬似コードにおける演算子の優先度＞

演算子の種類		演算子	優先度
式		()	高
単項演算子		not, +, −	↑
二項演算子	乗除	×, ÷	↑
	加減	+, −	↑
	関係	≠, ≦, ≧, <, >, =	↑
	論理積	and	↑
	論理和	or	低

1．プログラム中の空欄 ① から ⑥ に入れる正しい式を答えよ．

2．手続きsortを以下の配列listとその要素数7を引数として呼び出したとする．最初の呼び出しも含めて，手続きsortは何回呼び出されるか答えよ．

　　listの内容：{2, 8, 3, 5, 7, 4, 1}

3．手続きsort中の listの出力処理 でlistの内容を画面に出力する．例えば，listの内容が問2の時，{2, 8, 3, 5, 7, 4, 1}と出力される．問2のlistとその要素数7を引数としてsortを呼び出した際の出力の様子を答えよ．

4．手続きsortに与える配列の要素数をnとしたときの時間計算量をオーダー表記で答えよ．

2026年度　大学院外国人留学生入学試験問題
理工学研究科　前期課程　情報工学専攻
情　報　工　学

（注）問題番号または記号を必ず解答用紙に明記すること

問題III

左上端$(0,0)$、右上端$(W,0)$、左下端$(0,H)$、右下端(W,H)とする2次元整数座標$(0<W,H)$で表され、かつ、0（黒）または1（白）で表される2値画像を考える。このとき、以下の問に答えよ。

1. 画素(x_1,y_1)、(x_2,y_2)、(x_3,y_3)を頂点とする三角形の辺及び内部を画素値0で、それ以外の画素を画素値1で表現したい。全ての画素値を決定する手順を示せ。但し、$0<x_2<x_1<x_3<W$、$0<y_1<y_2<y_3<H$が成立している。手順の記述は文及び式、または、C言語等のプログラム形式等、形式を問わない。

2. 四近傍全てに隣接画素がある画素(x_0,y_0)に対して、以下の各式を全て満たす画素集合$\{(x_1,y_1)\}$、$\{(x_2,y_2)\}$、$\{(x_3,y_3)\}$がそれぞれ画像内でどのような領域となるか、違いを明確に説明せよ。

$$|x_1-x_0|+|y_1-y_0|\le r,$$
$$\sqrt{(x_2-x_0)^2+(y_2-y_0)^2}\le r,$$
$$\max\{|x_3-x_0|,|y_3-y_0|\}\le r$$

但し、$0<r<\min\{|W-x_0|,|x_0-W|,|H-y_0|,|y_0-H|\}$, r:整数

また、この結果を利用して、整数x、yに対する$|x|+|y|$、$\sqrt{x^2+y^2}$、$\max\{|x|,|y|\}$の大小関係を導け。

（注）問題番号または記号を必ず解答用紙に明記すること

問題Ⅰ. 次の英文を読み、英文の内容に従って問いに答えよ。

The distance between two pixels $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ on a two-dimensional digital image is defined as $d \geq 0$. It is known that there are multiple values for d that satisfy the mathematical definition of distance. Among these, the following three types are commonly used for two-dimensional digital images.

The first is called the Euclidean distance d_e, which is the shortest length between two points measured along a straight line—the distance we commonly use in daily life. It is expressed by the following formula:

$$d_e = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

The second is called the Manhattan distance d_m, which represents the distance when only vertical and horizontal movement is allowed, as if walking along grid-patterned streets. It is also known as the city block distance. It is expressed by the following formula.

$$d_m = |x_1 - x_2| + |y_1 - y_2|$$

The third is called the Chebyshev distance d_c, which counts diagonal movement as one step in addition to vertical and horizontal movement. Because it resembles the movement of a chess king, it is also called the king's distance. It is expressed by the following formula.

$$d_c = \max\{|x_1 - x_2|, |y_1 - y_2|\}$$

Among these three distances, Chebyshev distance is the most advantageous from a computational load perspective in computing.[a] Euclidean distance requires squaring, addition, and the highly costly calculation of square roots. In contrast, Manhattan distance can be computed using only absolute values and addition. Furthermore, Chebyshev distance only requires comparison operations to select the "larger" of two differences and does not require floating-point arithmetic. This makes it superior to the other two distances for fast computation in real-time demanding scenarios like image processing and pathfinding.

The distance between P_1 and P_2 varies depending on the type of the distance used. For example, when $P_1(2,1)$ and $P_2(6,4)$ are set, the distance between P_1 and P_2 are calculated as $d_e =$ (b), $d_m =$ (c), and $d_c =$ (d), respectively. Generally, the relationship of $d_c \leq d_e \leq d_m$ always holds for the same two points. When two pixels are sufficiently far apart, the equality in the above holds only when P_1 and P_2 are aligned vertically or horizontally.

Next, let's consider the shape formed by the set of pixels within a "distance of 10" from pixel $P_0(x_0, y_0)$ and the number of pixels contained within that shape. The shape representing $d_e \leq 10$ is the edge and interior of a circle centered at P_0 with a radius of 10. On the other hand, the shape representing $d_m \leq 10$ is the edge and interior of a 45-degree tilted square with a diagonal length of 20, where P_0 is the intersection point of the diagonals. Furthermore, the shape representing $d_c \leq 10$ is the edge and interior of a square with a side length of 20 and its center of gravity at P_0. As can be seen immediately by drawing the three resulting shapes,[e] the shape containing the most pixels internally is the Chebyshev distance, while the shape containing the fewest pixels is the Manhattan distance. That is, when the distance is sufficiently large, the relationship between the areas covered by distances of the same value is: area defined by (f) < area defined by (g) < area defined by (h).

1. 下線部（a）の記述の根拠となる説明を、本文内容に即して日本語で述べよ。
2. 文中（b）、（c）、（d）に整合する値をそれぞれ示せ。
3. 下線部（e）に整合する図を描け。図にはx軸、y軸を明記すること。
4. 文中（f）、（g）、（h）に整合する距離変数をそれぞれ解答せよ。
5. 挙げた3種の距離の名称のうち一つは地名、残り二つは人名に由来する。地名由来の距離はどれか。また、なぜそのような名称となったのか、本文内容に即して説明せよ。

（注）問題番号または記号を必ず解答用紙に明記すること

問題 II. SRT 除算アルゴリズム（段落①）と Pentium の FP 除算バグ（段落②〜⑤）に関する次の英文を読み，以下の設問に答えよ．Pentium は Intel が開発したマイクロプロセッサの名称である．また，FP は floating-point の略で，浮動小数点を意味する．

① The SRT division algorithm is commonly used in hardware for dividing floating-point numbers. It computes the quotient iteratively, selecting one quotient digit per iteration. In radix-2^n SRT division (e.g., radix-2 and radix-4), each iteration produces one quotient digit in base 2^n, which corresponds to n quotient bits. When $n > 1$, this digit is typically selected using a small lookup table. SRT algorithms are also used for square-root computation.

補足：「lookup table」は単にルックアップテーブルと訳せばよい．

② The error in the division portion of the floating-point unit of the Pentium microprocessor drew a lot of media attention late in 1994. An interesting artifact ⓐ of the error was that it ⓑ exposed the internals of the Pentium design to public scrutiny. Several individuals managed to reverse-engineer the implementation as a result of the design error. If the division unit had worked correctly, it would have been impossible to determine the nature of the implementation.

③ Intel uses a radix-4 SRT implementation for division in the Pentium. The error resulted from incorrect entries in the lookup table used to implement the quotient selection in the SRT algorithm. Five entries that should have had the value 2 had the value 0. According to Intel, the five entries were mistakenly omitted from the input script to the PLA (programmable logic array) used to generate the circuit layout for the table.

④ The error manifests itself only for a small minority of operands, but can be considerable when it appears. For example, one would expect the result of $r = x - \left(\frac{x}{y}\right) \times y$ to be zero. Calculating r with values $x = 4195835$ and $y = 3145727$ on a faulty Intel Pentium results in $r = 256$. The division $\frac{x}{y}$ in this case is accurate to only 14 bits rather than 53, i.e., worse than single precision for a double precision computation, and is an example of the worst-case error. The only inputs affected are those which at some time in the course of computation encounter the missing entries.

⑤ The bug affects only division and not square root, which does not use the table. In fact, this is one of the most intriguing aspects of this case. Ironically, the fateful quotient selection table actually appears over-designed, with higher resolution than necessary for radix-4 SRT division. Meanwhile, the Pentium implements square root using a radix-2 SRT algorithm, instead of implementing radix-4 square root as well.

出典：P. Soderquist and M. Leeser, Area and performance tradeoffs in floating-point divide and square-root implementations, *ACM Computing Surveys*, 28 (3), 518–564, 1996.（付録 A より抜粋，一部改変．）

1. 段落①を日本語に訳せ．
2. 段落②について下記の問に答えよ．
   (1) 「it」（下線部ⓑ）が指しているものを，この段落の英文から抜き出せ．
   (2) 下線部ⓐについて，なぜ「interesting」なのか日本語で説明せよ．
3. Pentium の FP 除算演算器において，なぜ除算の計算に誤りが発生するのか段落③から読み取って日本語で説明せよ．
4. 段落④について下記の問に答えよ．ただし $x = 4195835$, $y = 3145727$ とせよ．
   (1) この段落において，「$\frac{x}{y}$ の計算で発生する誤りの大きさ」は小さいと評価されているのか，大きいと評価されているのか答えよ．
   (2) 以下の［ア］〜［エ］の中から，「Pentium の FP 除算演算器で計算される $\frac{x}{y}$ の値」に最も近いものを選択せよ．（Pentium で計算される値との差の絶対値が最も小さくなるものを選択すればよい．）なお，$\frac{x}{y}$ の真の値は「1.3338204...」である．
   ［ア］1.333　　［イ］1.3338　　［ウ］1.3339　　［エ］1.334
5. 下記の項目［ア］〜［エ］の中から，誤りを含む項目を全て選択せよ．段落②〜⑤の内容に則して選択すること．
   ［ア］ Pentium の FP 除算演算器のバグは，どのような入力値に対しても除算の計算に影響を与える．
   ［イ］ Pentium の FP 除算演算器のバグは，除算の計算だけでなく平方根の計算についても影響を与える．
   ［ウ］ Pentium の FP 除算演算器は radix-4 STR アルゴリズムを使用している．
   ［エ］ Pentium の FP 平方根演算器は radix-4 STR アルゴリズムを使用している．

２０２６年度　中央大学大学院春季一般入学試験

理工学研究科　博士課程前期課程

＜情報工学専攻＞　情報工学

問題Ⅰは必答。

問題Ⅱ～Ⅴのうち、**２問**を選んで解答すること。

解答用紙上部の「問題番号」の枠内に、選択した問題番号（Ⅱ～Ⅴ）を記し、１枚の解答用紙につき１問解答すること。

上記事項が守られていない場合、解答が無効となることがありますので、注意してください。

（注）問題番号または記号を必ず解答用紙に明記すること

問題 I.　（全員解答すること）

1. 次の行列 A_n $(n \geq 3)$ を考える．

$$A_n = \begin{pmatrix} a & 0 & \cdots & 0 & -x_1 \\ 0 & a & \ddots & \vdots & -x_2 \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & a & -x_{n-1} \\ x_1 & x_2 & \cdots & x_{n-1} & a \end{pmatrix}$$

例えば，$n = 3$ の場合は次のようになる．

$$A_3 = \begin{pmatrix} a & 0 & -x_1 \\ 0 & a & -x_2 \\ x_1 & x_2 & a \end{pmatrix}$$

(1) $n = 3$ のときの行列式 $|A_3|$ を求めよ．

(2) $n = 4$ のときの行列式 $|A_4|$ を求めよ．

(3) $n \geq 4$ のときの行列式 $|A_n|$ を求めよ．

2. $f(x, y) = 3x^2 + 6xy + 3y^2 + x^3 + y^3$ の極値を求めたい．

(1) $f(x, y)$ が点 (a, b) で極値をとるならば，$f_x(a, b) = f_y(a, b) = 0$ であることを用いて，$f(x, y)$ が極値をとる候補点を全て求めよ．

(2) 点 (a, b) を $f(x, y)$ の極値の候補点とし，$D = f_{xx}(a, b)f_{yy}(a, b) - f_{xy}(a, b)^2$ とおくと，次の (i), (ii), (iii) が成り立つ．このことを用いて，(1) で求めた各点で極大値をとる，極小値をとる，極値をとらないのどれになるかを判定せよ．

(i) $D > 0$ のとき，$f_{xx}(a, b) > 0$ ならば $f(x, y)$ は (a, b) で極小値，$f_{xx}(a, b) < 0$ ならば $f(x, y)$ は (a, b) で極大値をとる．

(ii) $D < 0$ のとき，$f(x, y)$ は (a, b) で極値をとらない．

(iii) $D = 0$ のとき，$f(x, y)$ が (a, b) で極値をとるかどうかは D の値だけでは判定できない．

２０２６年度　大学院春季入学試験問題
理工学研究科　前期課程　情報工学専攻
情　報　工　学

（注）問題番号または記号を必ず解答用紙に明記すること

問題 IL つぎの設問に答えよ.

1. 画面に表示される n 枚の色つき板 (panel) を考える ($1 \leq n < 10^5$). 板は番号 k ($0 \leq k < n$) で指定され, その色は色番号 C_k ($0 \leq C_k < n$) で示される. ここで, 板 i と j ($0 \leq i,j < n$) について, 板 j の色番号 C_j と同じ色番号を持つすべての板の色番号を, C_i に上書きする操作を「set_color(i,j)」と記す. 初期状態では $C_k = k$ ($0 \leq k < n$) とし, m 個の操作を順次施し, 各板の色番号を計算する. 色は複数回上書きされ得る.

```
1:  #include<stdio.h>
2:  int n; int c[100000];
3:  int get_color(int i){ return c[i];} /* 板 i の色番号を返す関数 */
4:  void set_color(int i,int j){int cj=c[j],k; for(k=0;k<n;k++)if(c[k]==cj)c[k]=c[i];}
5:  void put_color(){for(int i=0;i<n;i++)printf("%d%c",get_color(i),(i<n-1)?' ':'\n');}
6:  int main(){ int m;
7:    scanf("%d %d",&n,&m); /* n: 板数, m: 指示数 */
8:    for(int k=0;k<n;k++) c[k]=k;
9:    for(int k=0;k<m;k++){int i,j; scanf("%d %d",&i,&j); set_color(i,j); put_color();}
10: }
```

図 1　プログラム P1. C_k の値を配列要素 c[k] に計算する. (紙面の都合上エラー処理等は省略して記載.)

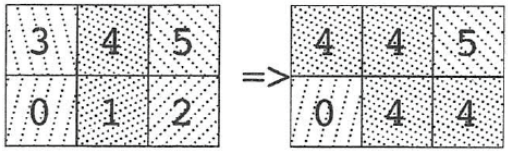


図 2　初期状態（左）と最終状態（右）. 数値は色番号を表す. (色をパターンで示した.)

上記 P1 は C_k を計算するプログラムの例である. まず, データの先頭行から n と m の値を読込む. 続いて m 行の各行に記された 2 個の整数「i　j」を読み込み, 各行が表す「set_color(i,j)」という操作を順番に施す.

例えば, 1 行に整数を 2 個ずつ「6　3」,「4　1」,「2　3」,「4　3」として, 全 4 行をこの順に与えた場合を考える. 入力の先頭行から $n = 6$, $m = 3$ が設定され, P1 の 8 行目で配列 c[] の先頭 6 個の要素が {0,1,2,3,4,5} に初期化される. set_color 操作を表すデータは 3 ($= m$) 行あり, 次の行「4　1」が 1 番目の操作「set_color(4,1)」を表し, その結果として「0 4 2 3 4 5」が出力され,「2　3」が 2 番目の操作を表しその結果が「0 4 2 2 4 5」,「4　3」が 3 番目の操作を表しその結果が「0 4 4 4 4 5」と表示される (図 2).

以下の設問 (1), (2) に答えよ.

(1) このプログラム P1 を実行し, 入力として各行に 2 個の整数を, 順番に「8　5」,「5　3」,「2　4」,「6　0」,「4　5」,「7　3」として, 全 6 行を与えたときの出力を記せ.

(2) 上記 P1 では, 配列「int c[100000];」に色番号を格納している. プログラム P1 の 3~4 行目に基づき, 関数 get_color() と set_color() の計算方法を説明し, 正しい結果が得られることを説明せよ.

2. 図 1 のプログラム P1 中の配列 c[i] の値が表す意味を変更し, 板 i の色番号と同じ色番号を持つ板の番号 (のひとつ) を表すこととする. 次の方針により, 上記と異なるアルゴリズムを構成できる.

(方針 1) 初期状態では, 番号 k の板は色番号 k の代表板であり, c[k] の値は k である.

(方針 2) ある板 i の代表板の色番号は, c[i] の値を j として, $j = i$ なら i である. そうでなければ, 板 j の代表板の色番号に等しい. c[i] の値はその代表板の番号 (色番号と等しい) に更新する.

(方針 3) 色の上書き指示「set_color(i,j)」の処理は, 上書きされる板 j の代表板の色番号を, 上書きする板 i の代表板の色番号にする.

以下の設問 (1), (2) に答えよ.

(1) 上記 P1 の 3 行目の「int get_color(int i){...}」を再帰関数として再定義せよ. それに合わせて, 4 行目の「void set_color(int i,int j){...}」を再定義せよ. 各定義は 3 行以下で記述し, 処理の長所・短所について P1 と比較し説明せよ. この修正版を P2 と記す. (エラー処理等の記述は不要.)

(2) (1) の修正版 P2 に, 問題 **1**(1) と同じ 6 行の入力を与えたときの出力を記し, その動作を説明せよ.

問題 III.

以下の性質が成り立っているような離散型確率変数 X を考える: 任意の $k \in \mathbb{N} = \{1, 2, 3, \ldots\}$ に対し,

$$\Pr(X = k) = \frac{0.999^{k-1}}{1000}.$$

1. $\Pr(X > k)$ を $k = 0, 1, 2, \ldots$ に対し求めよ.
2. X の期待値 $\mathbf{E}[X]$ を求めよ.
3. $\Pr(X > 1001 \mid X > 1000)$ を求めよ.

今, ある問題 Π と, それを解くアルゴリズムの設計を考える.

4. 問題 Π に対し, 計算時間 T 以内に正しい解を確率 δ で出力するアルゴリズム A が存在するとする $(0 < \delta \leq 1)$. また, 出力の正当性は t 時間以内で検証できるとする. ここで, 検証手続きは常に正しく, 正しい解は必ず受理され, 誤った解は必ず棄却されるものとする. 問題 Π に対し, 確率 1 で正しい解を出力するアルゴリズム B を (A を用いて) 設計せよ. また B の計算時間の期待値 (の上界) を求めよ.
5. Π は最大化問題であると仮定する. 即ち, 各入力に対して実数値を返す目的関数が与えられており, その最大値を求める問題である (例: ナップサック問題, 最大カット問題, ...).
   問題 Π に対し, 計算時間 T 以内に確率 ϵ $(0 < \epsilon < 1)$ で最大値を返すアルゴリズム A が存在するとする (ここで, 残りの確率 $1 - \epsilon$ では最大値でない実数値を出力するとする). 任意の変数 $0 < \delta < 1$ に対し, 最大値を $1 - \delta$ 以上の確率で返すアルゴリズム B を (A を用いて) 設計せよ. また B の計算時間の上界を求めよ.

問題IV.

グラフ$G=(V,E)$の頂点に色を塗ることを考える．k個の色を用いて，どの辺の両端点も互いに異なる色で塗り分けるとき，Gはk彩色可能であるという．Gの頂点の彩色に必要なkの最小値を彩色数とよび，$\chi(G)$で表す．

図1は3彩色の例である．頂点aから頂点dの○の中にある数字が色を表す．隣接する頂点には異なる番号（色）が割り当てられていることが確認できる．この例では，隣接していない頂点bとdには同じ番号（同じ色）が割り当てられている．

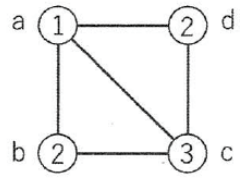


図1　3彩色の例

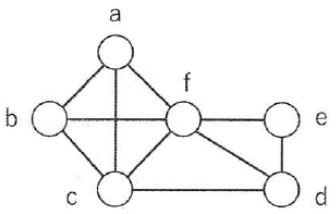


図2　6頂点のグラフ

1. 図2のグラフの頂点をできるだけ少ない色で彩色せよ．解答欄には図2のグラフを描き，グラフの頂点を表す○の中に，色に対応する番号を書いて解答すること．

2. 頂点vに接続する辺の数をその頂点の次数とよび，d_vで表す．以下は頂点彩色のアルゴリズムである．

彩色アルゴリズム

ステップ1．頂点を次数の大きい順に並べ，$v_1,v_2,\ldots,v_n$とおく（つまり$d_{v_1}\geq d_{v_2}\geq\cdots\geq d_{v_n}$が成り立つ）．次数が同じ頂点はどちらを先にしてもよい．

ステップ2．$v_1,v_2,\ldots,v_n$の順に頂点を選び，各v_iに割り当て可能なもっとも小さい番号の色（v_iに隣接するv_1からv_{i-1}の頂点に使用されていない番号で最小のもの）を割り当てる．

彩色アルゴリズムを図2のグラフに適用して得られる頂点の彩色を答えよ．解答欄には図2のグラフを描き，グラフの頂点を表す○の中に，色に対応する番号を書いて解答すること．ステップ1で定めた頂点の順序も答えよ．

3. 彩色アルゴリズムは最小の色数で彩色するとは限らない．このアルゴリズムが$\chi(G)$個の色で彩色しないグラフGの例（グラフとステップ1の頂点の順序）を述べ，アルゴリズムが$\chi(G)$個の色で彩色しない理由を説明せよ．

4. 彩色アルゴリズムは彩色数$\chi(G)$の上界を与えている．この上界を理由とともに答えよ．数式で記述してもよいし，文章で説明してもよい．

5. 彩色数$\chi(G)$の下界として適切なものを答えよ．例などを使ってわかりやすく説明すること．

問題 V.

1. ある工場では製品 A と製品 B を生産している．これらの製品はどちらも材料 1, 材料 2, 材料 3 から作られる．それぞれの製品を 1kg 生産するために必要な材料の量は下の表の通りである．

	材料 1 の使用量 (kg)	材料 2 の使用量 (kg)	材料 3 の使用量 (kg)
製品 A	2	25	20
製品 B	12	50	20

製品 A を 1kg 生産すると利益が P_A 円，製品 B を 1kg 生産すると利益が P_B 円得られる．また，材料 1 は M_1kg，材料 2 は M_2kg, 材料 3 は M_3kg だけ利用可能である．それぞれの製品の生産量は必ずしも整数である必要はない．このとき，利益を最大化するために製品 A と製品 B をそれぞれいくら生産すればいいかを求めたい．

(1) この問題を表わす線形計画問題を与えなさい．ただし，問題の変数は，製品 A の生産量 (kg) を表す変数 x_A と，製品 B の生産量 (kg) を表す変数 x_B のみとすること．

(2) 問題のパラメータが次の値を取るとき，利益を最大化する製品 A と製品 B の生産量を求めなさい．

$$P_A = 140, \quad P_B = 210, \quad M_1 = 120, \quad M_2 = 600, \quad M_3 = 320$$

2. ある一直線上に伸びた通りに沿って 4 軒の家が建っている．それぞれの座標を p_1, p_2, p_3, p_4 とする．この通りにゴミ捨て場を 1 カ所設置する．ゴミ捨て場を設置できる場所の座標は L 以上 R 以下に限られている．すべての家のゴミ捨て場までの距離を合計したものを最小化するように，ゴミ捨て場の設置場所を決定したい．

設置するゴミ捨て場の座標を求める問題を表現する線形計画問題を与えなさい．変数は自由に導入していいが，導入した変数の最適解における値とゴミ捨て場の座標がどのように対応するかも説明すること．

3. 次の線形計画問題を考える．

$$\begin{aligned} \text{maximize} \quad & 10x_1 + 25x_2 + 5x_3 + 36x_4 \\ \text{subject to} \quad & 8x_1 - x_2 + 3x_3 + 4x_4 = 11, \\ & 5x_1 + 6x_2 + 2x_3 + 10x_4 \leq 40, \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

(1) この線形計画問題の双対問題を与えなさい．

(2) $(x_1, x_2, x_3, x_4) = (0, 4.9, 5.3, 0)$ は，この線形計画問題の最適解である．このことを証明しなさい．