大学院情報理工学研究科　博士前期課程：一般入試（2025 年 8 月 19 日実施）

情報学専攻　必須問題　解答例

必須問題では、基礎的な知識や理解を問う問題を出題しました。解答が数式または数値で明記できるものについて、その一例を以下に示しますが、これと同等な他の表現もありえます。

「線形代数」および「微分積分」

［1］

(1) $\begin{bmatrix} -4 \\ 1 \\ 5 \\ -2 \end{bmatrix}$

(2) $-1, 3$

(3) $\alpha = \beta = \gamma = -1$

［2］

(1) (i) $y = 2x$

(ii) $f(x, y)$ は点 $(1, 1)$ で極小値 -1 をとる.

(2) $\dfrac{1}{45}$

以上

大学院情報理工学研究科　博士前期課程：一般入試（2025 年 8 月 19 日実施）

情報学専攻　選択問題　解答例

選択問題では、専門的な知識や理解を問う問題を出題しました。解答については、その一例を以下に示しますが、これと同等な他の表現もありえます。

1　アルゴリズムとデータ構造

プログラムコード 1 は最小値ヒープを取り扱うための C 言語によるコードである．コード全体を出すと下記のようなコードである．

```
void SWAP(int *a, int *b)
{
  int t = *a; *a = *b; *b = t;
}

void min_sift_up(int i, int h[])
{
  while(i > 0){
    int p = (i-1) / 2;
    if(h[p] < h[i])      break;
    SWAP(&h[p], &h[i]);
    i = p;
  }
}

void min_sift_down(int i, int h[], int sz)
{
  while( 1 ){
    int l = 2*i+1, r = 2*i+2, j = i;
    if(l < sz && h[l] < h[j])      j = l;
    if(r < sz && h[r] < h[j])      j = r;
    if(j == i)      break;
    SWAP(&h[j], &h[i]);
```

```
int min_sz = 0;
int min_h[K]; // K はヒープの大きさ

void min_push(int x)
{
  if(min_sz >= K)                return;
  min_h[min_sz] = x;
  min_sift_up(min_sz++, min_h);
}

int min_pop()
{
  int ret = min_h[0];
  min_h[0] = min_h[--min_sz];
  min_sift_down(0, min_h, min_sz);
  return ret;
}

void min_push_wrapper(int x)
{
  if(min_sz < K)      min_push(x);
  else if(x > min_h[0]){
    min_h[0] = x;
```

プログラムコード 1

最小値ヒープは，親ノードの値が，子ノードの値より小さいという特性を持ち，値ノードにはヒープ内の最小値が配置される．この特性を維持するために下記の関数群を定義している．

- void min_sift_up(int i, int h[])：i 番目のノードからボトムアップでヒープを維持する関数
- void min_sift_down(int i, int h[], int sz)：i 番目のノードからトップダウンでヒープを維持する関数
- void min_push(int x)：ヒープに値 x を保持するための関数
- int min_pop()：ヒープから最小値を取り出し，ヒープ構造を維持する関数
- void min_push_wrapper(int x)：最大 K 個の値をヒープに維持し，ヒープ内の最小値よりも大きい値が x として与えられた場合に，最小値を捨て新たな x を保持する関数

1. 空欄 A-C に関しては上記コードを参照のこと．別表現でも可能

A: h[p] < h[i]

B: h[l] < h[j]

C: h[r] < h[j]

2. この小問では K=8 の場合を取り扱うので最大でも 8 個までしか値が保持できないことに注意する．

A1: 2, 4, 5, 9, 12, 7, 8, 10

A2: 2, 7, 4, 8, 12, 9, 6, 11

3. この小問では，ヒープから最小値を取り出した場合に，ヒープを維持するのに必要な時間計算量を問うている．ヒープが完全二分木を構成することに鑑みて，入れ替え回数が最大でも完全二分木の高さに従うことを注意する．

時間計算量は $O(log\ K)$

4. 空欄 D に関しては上記コード参照．別表現でも可能

min_h[0] = x;

min_sift_down(0, min_h, min_sz)

5. ここでは，長さ K の最小値ヒープに，N 個の系列データを与えて，系列データ中の大きいもの K 個を保持する．一つのデータが与えられたときの計算量は，最大でも木の高さ分の入れ替えしか起こらない．また，この計算は N 回繰り返されることに注意する．

時間計算量は $O(N\ log\ K)$

6. 上記問題 5 に対して具体的な値 $N = 10^6$,$K = 10^2$ を与えた場合の評価を行う．比較対象はクイックソートであり，クイックソートにかかる計算時間は $O(N\ log\ N)$ であることに注意する．数値を与えると

$$N\ log\ N = 10^6 \times 6$$
$$N\ log\ K = 10^6 \times 2$$

となり，およそ $log\ N\ /\ log\ K = 3$ 倍クイックソートの方が計算時間かかることになる．

クイックソートの方が 3 倍程度遅い

さらにプログラムコード２を考える．

```
int min_sz=0, max_sz=0;
int min_h[K], max_h[K]; // K はヒープの大きさ

void median_push(int x)
{
  if(max_sz == 0 || x <= max_h[0]){
    max_push(x);
    if(max_sz > min_sz+1)
      min_push(max_pop());
  }
  else{
    min_push(x);
    if(min_sz > max_sz)
      max_push(min_pop());
```

```
double get_median()
{
  double ret;
  if((min_sz + max_sz) % 2 == 0)
    ret = min_h[0] + max_h[0] / 2.0;
  else
    ret = max_h[0]                    ;
  return ret;
}
```

プログラムコード２

このコードに加えて，最小値ヒープと対となる最大値ヒープを取り扱うための関数群　max_push(),

```
void max_sift_up(int i, int h[])
{
  while(i > 0){
    int p = (i-1) / 2;
    if(h[p] > h[i])      break;
    SWAP(&h[p], &h[i]);
    i = p;
  }
}

void max_sift_down(int i, int h[], int sz)
{
  while( 1 ){
    int l = 2*i+1, r = 2*i+2, j = i;
    if(l < sz && h[l] > h[j])      j = l;
    if(r < sz && h[r] > h[j])      j = r;
    if(j == i)      break;
    SWAP(&h[i], &h[j]);
    i = j;
```

```
void max_push(int x)
{
  if(max_sz >= K)                return;
  max_h[max_sz] = x;
  max_sift_up(max_sz++, max_h);
}

int max_pop()
{
  int ret = max_h[0];
  max_h[0] = max_h[--max_sz];
  max_sift_down(0, max_h, max_sz);
  return ret;
}
```

プログラムコード３

max_pop() を考える．これらの実装例はプログラムコード３のような例となる．

プログラムコード１～３で取り扱っている最小値ヒープと最大値ヒープを用いて，与えられた系列の中央値を計算するコード　median_push()を考える．操作としては最大値ヒープの大きさ　max_sz　が min_sz　と等しいか１だけ大きくなるように調整しながら，ヒープの出し入れを行うことで，最大値ヒープと最小値ヒープの値ノードを維持する．系列長が 2M+1　の奇数個であれば，系列中の値の大きな M 個は最小値ヒープに保持される（問題５参照）．逆に値の小さい M+1 個は，最大値ヒープに保持されることになる．このとき最大値ヒープの根ノードは値の小さいものの中での最大値となり，これより値の大きなものは最小値ヒープに保持されていることとなるため，この値が中央値となる．一方，系列長が偶数の 2M であれば，最大値ヒープと最小値ヒープの中間の値が中央値となる．

7. コードに従って順次値を入れていく．
   min_h: 5, 8, 6
   max_h: 4, 3, 2, 1

8. median_push() では最大値ヒープと最小値ヒープの大きさがほぼ等しくなるように制御していることに注意する．
   min_h の要素数：M
   max_h の要素数：M+1

9. プログラムコード２参照．与えられた系列長が奇数個であれば，最大値ヒープの最小値が中央値となり，偶数個であれば最大値ヒープと最小値ヒープの中間値が中央値なることに注意する．
   E: (min_h[0] + max_h[0])/ 2.0;
   F: max_h[0];

2　確率・オペレーションズリサーチ

問 1

問 1-1：期待値 0，標準偏差 1

問 1-2：期待値 0，標準偏差 $\sqrt{n}$

問 1-3：0.00135

注：Y_{10000} は整数値を取るため，$Y_{10000} \geqq 301$と変形した近似により得られる 0.00131 も正解とした．

問 1-4：$a = 1/2$

問 1-5：独立でない．$Z_2 = 1$のとき$Z_1 - Z_2 = 0$であるが，$Z_2 = 0$のとき$Z_1 - Z_2 \neq 0$であり，Z_2の値によって$Z_1 - Z_2$の条件付き分布が異なる．

問 2

問 2-1：D：指数分布

問 2-2：期待値$1/\lambda$，分散$1/\lambda^2$．

問 2-3：$\lambda/(\lambda - t)$

問 2-4：$e^{-\lambda a} - e^{-\lambda b}$

問 2-5：どちらも$1 - e^{-\lambda(b-a)}$となり等しい（無記憶性）

問 3

問 3-1

$$\max\ 600x_1 + 500x_2 + 1000x_3$$
$$\text{s.t}\ x_1 + x_2 + x_3 \leq 10,$$
$$25x_1 \leq 90$$
$$40x_2 \leq 140,$$
$$30x_3 \leq 120,$$
$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0.$$

問 3-2

$$x_1 + x_2 + x_3 + x_4 = 10,$$
$$25x_1 + x_5 = 90$$
$$40x_2 + x_6 = 140,$$
$$30x_3 + x_7 = 120,$$
$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0, x_6 \geq 0, x_7 \geq 0$$

問 3-3

初期表	c_i	600	500	1000	0	0	0	0	
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	定数項
0	x_4	1	1	1	1	0	0	0	10
0	x_5	25	0	0	0	1	0	0	90
0	x_6	0	40	0	0	0	1	0	140
0	x_7	0	0	30	0	0	0	1	120
	π_i	0	0	0	0	0	0	0	
	$c_i - \pi_i$	600	500	1000	0	0	0	0	

更新1回目	c_i	600	500	1000	0	0	0	0	
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	定数項
0	x_4	1	1	0	1	0	0	−1/30	6
0	x_5	25	0	0	0	1	0	0	90
0	x_6	0	40	0	0	0	1	0	140
1000	x_3	0	0	1	0	0	0	1/30	4
	π_i	0	0	1000	0	0	0	100/3	
	$c_i - \pi_i$	600	500	0	0	0	0	−100/3	

更新2回目	c_i	600	500	1000	0	0	0	0	
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	定数項
0	x_4	0	1	0	1	−1/25	0	−1/30	12/5
600	x_1	1	0	0	0	1/25	0	0	18/5
0	x_6	0	40	0	0	0	1	0	140
1000	x_3	0	0	1	0	0	0	1/30	4
	π_i	600	0	1000	0	24	0	100/3	
	$c_i - \pi_i$	0	500	0	0	−24	0	−100/3	

更新3回目	c_i	600	500	1000	0	0	0	0	
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	定数項
500	x_2	0	1	0	1	−1/25	0	−1/30	12/5
600	x_1	1	0	0	0	1/25	0	0	18/5
0	x_6	0	0	0	−40	8/5	1	4/3	44
1000	x_3	0	0	1	0	0	0	1/30	4
	π_i	600	500	1000	500	4	0	50/3	7360
	$c_i - \pi_i$	0	0	0	−500	−4	0	−50/3	

更新4回目	c_i	600	500	1000	0	0	0	0	
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	定数項
	π_i								
	$c_i - \pi_i$								

最適解（x_1^*, x_2^*, x_3^*）：$(\frac{18}{5}, \frac{12}{5}, 4)$

最適解における目的関数の値（z^*）：7360

3 離散数学

問 1.

(1) 1 ⑤

(2) 2 ①

(3) 3 ④

(4) 4 ④

問 2.

(1) 5 ② 6 ⑥ 7 ⑥

(2) 8 ④ 9 ⑥

(3) 10 ① 11 ⑧

問 3.

12 ④ 13 ⑧ 14 ⓪ 15 ① 16 ① 17 ⓪ 18 ① 19 ③ 20 ⑨
21 ⑤ 22 ① 23 ③

問 4.

(1) $\forall(x_1, x_2), \forall(x_1', x_2') \in X^2, (x_1, x_2) \neq (x_1' x_2') \Rightarrow \tilde{f}\big((x_1, x_2)\big) \neq \tilde{f}\left((x_1', x_2')\right)$

(2) 81

(3) 存在しない

問 5.

(1)

- $X = 1$のとき，1ターン目で先手が1個石を取って勝利.
- $X = 4$のとき，1ターン目で先手が1個石を取る．残りの石の個数は3である．2ターン目で後手が石を0～2個取ることができる．残りの石の個数は1～3のいずれかである．3ターン目で先手が石を全て取って勝利.
- $X = 9$のとき，1ターン目で先手が1個石を取る．残りの石の個数は8である．2ターン目で後手が石を0～2個取ることができる．3ターン目で先手は，これまでに両者が取った石の合計が4個となるように石を取る．残りの石は5個である．4ターン目で後手が石を0～4個取ることができる．残りの石の個数は1～5のいずれかである．5ターン目で先手が石を全て取って勝利.

(2)

(i) $k = 1$のとき，1ターン目で先手が石を1個取って勝利.

4　計算機工学

1. 計算機アーキテクチャ分野における数値の取り扱いなどに関する基礎を問う．

(1) $(1045)_{10}$

(2) 和ビット：$S = P \oplus C_{in}$ あるいは $S = \bar{P} \cdot C_{in} + P \cdot \overline{C_{in}}$

キャリー出力：$C_{out} = G + P \cdot C_{in}$

(3) $(00110011)_2$

2. キャッシュのヒット率に関する基礎を問う．

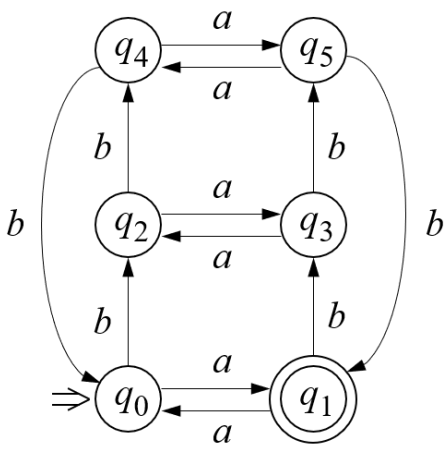
(1) $1 + 0.04 \times 20 = 1.8\,ns$

(2) $1ns +$ キャッシュミス率$\times 20ns \leq 2ns$ ⇒ キャッシュミス率 $\leq 5\%$　⇒　最大ミス率は 5%

3.　決定性有限オートマトンとそれが受理する正則言語に関する基礎を問う．

(1) *aaa*, *babb*, *ababba*, *bbaaab*

(2) 下図のとおり．



(3)　$\delta(q_0, a)=q_1$, $\delta(q_0, b)=q_2$, $\delta(q_1, a)=q_0$, $\delta(q_1, b)=q_3$, $\delta(q_2, a)=q_3$, $\delta(q_2, b)=q_4$,
$\delta(q_3, a)=q_2$, $\delta(q_3, b)=q_5$, $\delta(q_4, a)=q_5$, $\delta(q_4, b)=q_0$, $\delta(q_5, a)=q_4$, $\delta(q_5, b)=q_1$.
$F=\{q_1\}$

4.　文脈自由文法とそれが生成する文脈自由言語，および言語の演算に関する基礎を問う．

(1) $L(G_1)=\{a^i b \mid i \geqq 0\}$，$L(G_2)=\{a^i b^i \mid i \geqq 1\}$

(2) *bab*, *abab*, *aabab*, *baabb*, *abaabb*

(3) $S_3 \rightarrow S_1 S_2$