情報工学

予想問題　第7回　解答

（ハードウェア設計重視年）

記述式の小問については解答例を示す。同じ内容が伝われば表現は問わない。

作成：Claude（Anthropic）

1 【必須問題】アルゴリズムとプログラミング

(1-1) 満たそうとしている性質：**すべての節点について，その節点の値が両方の子の値以上である**（最大ヒープ，max-heap の性質）。

空欄 (a)：`n / 2 - 1`

（$i \geq n/2$ の節点は葉であり子を持たないため `down` を呼ぶ必要がない。子を持つ最後の節点は添字 $\lfloor n/2 \rfloor - 1$ である。）

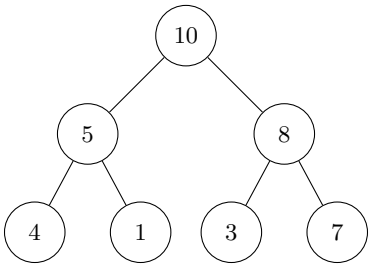
(1-2) 初期配列 $A = [4, 10, 3, 5, 1, 8, 7]$，$n = 7$ である。$i = 2, 1, 0$ の順に `down` を呼ぶ。

i	操作	操作後の A
2	$A[2] = 3$ と子 $A[5] = 8, A[6] = 7$ を比較。最大は 8。交換。	$[4, 10, 8, 5, 1, 3, 7]$
1	$A[1] = 10$ と子 $A[3] = 5, A[4] = 1$ を比較。10 が最大。交換なし。	$[4, 10, 8, 5, 1, 3, 7]$
0	$A[0] = 4$ と子 $A[1] = 10, A[2] = 8$ を比較。最大は 10。交換。	$[10, 4, 8, 5, 1, 3, 7]$
	続いて $i = 1$：$A[1] = 4$ と子 $A[3] = 5, A[4] = 1$。最大は 5。交換。	$[10, 5, 8, 4, 1, 3, 7]$

よって `build` 完了時の配列は

$$A = [10,\ 5,\ 8,\ 4,\ 1,\ 3,\ 7].$$

2 分木として図示すると次のとおり。



(2-1) `build` 後の出力は A の内容である。次に `extract` を 3 回呼ぶと，最大値が順に取り出される。`count` は `down` の中で実際に交換が起きた回数である。内訳は以下のとおり。

- `build`：$i = 2$ で 1 回（3 と 8 の交換），$i = 1$ で 0 回，$i = 0$ で 2 回（4 と 10，続いて 4 と 5 の交換）。計 3 回。
- 1 回目の `extract`：根に移した 7 が 8 と交換されて 1 回。
- 2 回目の `extract`：根に移した 3 が 7 と交換されて 1 回。
- 3 回目の `extract`：根に移した 1 が 5 と交換され，さらに 4 と交換されて 2 回。

合計 $3 + 1 + 1 + 2 = 7$ である。出力は以下のとおり。

```
10 5 8 4 1 3 7
10 8 7
7
```

(2-2) **1 回目の `extract`**：$A[0] = 10$ を取り出し，末尾 $A[6] = 7$ を根に移して $n = 6$ とし `down(0)` を実行する。

7 と子 5,8 を比較 → 8 と交換（添字 2 へ）。続いて 7 と子 $A[5] = 3$ を比較 → 交換不要。よって

$$A = [8,\ 5,\ 7,\ 4,\ 1,\ 3] \qquad (n = 6).$$

2 回目の `extract`：$A[0] = 8$ を取り出し，末尾 $A[5] = 3$ を根に移して $n = 5$ とし `down(0)` を実行する。

3 と子 5,7 を比較 → 7 と交換（添字 2 へ）。添字 2 に子はないので終了。よって

$$A = [7,\ 5,\ 3,\ 4,\ 1] \qquad (n = 5).$$

(3-1) 最悪時間計算量は $\mathbf{O}(\log \mathbf{n})$。

根拠：`down` は，交換が起きるたびに着目する節点を子へ 1 段降ろす。要素数 n の 2 分ヒープは完全 2 分木に近い形をしており，その高さは $\lfloor \log_2 n \rfloor$ である。したがって降下できる段数は高々木の高さであり，各段での処理（2 個の子との比較と交換）は定数時間であるから，全体で $O(\log n)$ となる。

(3-2) `build` では，深さの深い節点ほど個数が多く，かつそこから降下できる段数は少ない。高さ h の位置にある節点は高々 $\lceil n/2^{h+1} \rceil$ 個であり，そこからの降下は高々 h 段である。よって総作業量は

$$\sum_{h=0}^{\lfloor \log_2 n \rfloor} \left\lceil \frac{n}{2^{h+1}} \right\rceil \cdot O(h) = O\left(n \sum_{h=0}^{\infty} \frac{h}{2^{h+1}} \right) = O(n)$$

となる（$\sum h/2^h$ が定数に収束するため）。

一方，1 個ずつ挿入する方法では，k 番目の要素を挿入するたびに葉から根に向かって高々 $\log_2 k$ 段上昇する。n 個すべてについて総和をとると $\sum_{k=1}^{n} \log_2 k = O(n \log n)$ となる。

要するに，`build` は「多数の節点が浅くしか動かない」のに対し，逐次挿入は「多数の節点が深くまで動きうる」ため差が生じる。

(3-3) 方法：まず `build` により最大ヒープを構成する。次に `extract` を n 回繰り返し，取り出した値を順に並べる。最大値から順に取り出されるので，得られる列は降順に整列されている。

（配列内で行う場合は，根と末尾を交換してヒープの大きさを 1 減らし `down(0)` を実行する操作を繰り返せばよい。この場合は追加の記憶領域を必要としない。）

時間計算量：`build` が $O(n)$，`extract` 1 回あたり $O(\log n)$ でそれを n 回行うので

$$O(n) + O(n \log n) = \mathbf{O}(\mathbf{n} \log \mathbf{n}).$$

2 【必須問題】計算機システムとシステムプログラム

(1-1) 1 セットあたりの容量は $64 \times 4 = 256$ バイトであるから，セット数は

$$\frac{16\,\mathrm{KiB}}{256\ \text{バイト}} = \frac{16384}{256} = \textbf{64 セット}.$$

ビット数の内訳：

$$\text{オフセット} = \log_2 64 = \textbf{6 ビット}, \qquad \text{インデックス} = \log_2 64 = \textbf{6 ビット},$$

$$\text{タグ} = 32 - 6 - 6 = \textbf{20 ビット}.$$

(1-2) AMAT の定義より

$$\text{AMAT} = \text{ヒット時間} + \text{ミス率} \times \text{ミスペナルティ} = 1 + 0.03 \times 80 = 1 + 2.4 = \textbf{3.4 サイクル}.$$

(1-3) 1 次キャッシュでミスした場合，L2 へアクセスする（10 サイクル）。さらに L2 でもミスした場合（確率 25 %）に主記憶へアクセスする（80 サイクル）。よって 1 次キャッシュのミスペナルティは

$$10 + 0.25 \times 80 = 10 + 20 = 30 \text{ サイクル}$$

である。したがって

$$\text{AMAT} = 1 + 0.03 \times 30 = 1 + 0.9 = \textbf{1.9 サイクル}.$$

効果：2 次キャッシュの導入により 3.4 サイクルから 1.9 サイクルへとおよそ 44 % 短縮された。1 次キャッシュでミスしたアクセスの大半（75 %）を，主記憶より格段に高速な L2 で処理できるためである。

(1-4) ブロックサイズを 256 バイトにすると，1 セットあたりの容量は $256 \times 4 = 1024$ バイトとなるので

$$\text{セット数} = \frac{16384}{1024} = \textbf{16 セット}, \qquad \text{インデックス} = \log_2 16 = \textbf{4 ビット}.$$

（オフセットは $\log_2 256 = 8$ ビットとなり，タグは $32 - 8 - 4 = 20$ ビットで変わらない。）

長所：一度のミスでより広い範囲のデータを取り込むため，空間的局所性 (spatial locality) を活用でき，連続したアドレスへのアクセスに対するミス率が下がる。

短所：同じ容量ではブロック数（セット数）が減るため，離れたアドレスどうしの競合性ミス (conflict miss) が増えやすい。また 1 回のミスで転送するデータ量が増えるためミスペナルティ自体が大きくなる。

(2-1) 以下の実行順序でデッドロックが発生する。

1. P_1 が `P(s1)` を実行し，$s_1 = 0$ となって資源 R_1 を獲得する。
2. ここで P_2 に切り替わり，P_2 が `P(s2)` を実行し，$s_2 = 0$ となって資源 R_2 を獲得する。
3. P_2 が続けて `P(s1)` を実行するが $s_1 = 0$ であるため待ち状態に入る。
4. P_1 に戻り，P_1 が `P(s2)` を実行するが $s_2 = 0$ であるため待ち状態に入る。

この時点で P_1 は P_2 が保持する R_2 の解放を待ち，P_2 は P_1 が保持する R_1 の解放を待っている。どちらも先へ進めず，永久に待ち続ける。

(2-2) デッドロックの 4 条件は以下のとおり。

- 相互排除 (mutual exclusion)：資源は同時に一つのプロセスしか使用できない。
- 保持と待機 (hold and wait)：資源を保持したまま別の資源の解放を待つ。
- 横取り不可 (no preemption)：他のプロセスが保持する資源を強制的に奪えない。

- 循環待機 (circular wait)：資源の待ち関係に循環が存在する。

修正例：P_2 の資源獲得順を P_1 と同じ「s_1 を先に，s_2 を後に」へ変更する。

```
プロセス P2（修正後）
P(s1);
P(s2);
  /* R1 と R2 を使用 */
V(s2);
V(s1);
```

すべてのプロセスが資源に付けた番号の昇順にのみ獲得するため，「P_1 が s_2 を待ち P_2 が s_1 を待つ」という待ちの循環が生じ得なくなる。すなわち**循環待機**の条件を崩している。

(2-3) 各プロセスがあと必要とする資源数（Need = 最大要求数 − 現在の割り当て数）は

プロセス	A	B	C
P_1	7	2	3
P_2	1	2	2
P_3	2	0	0

利用可能な資源は $(3,3,2)$ である。順に調べる。

1. P_2 の Need $(1,2,2) \le (3,3,2)$ であるから P_2 を実行できる。完了後，割り当て $(2,0,0)$ が返却され利用可能は $(5,3,2)$ となる。
2. P_3 の Need $(2,0,0) \le (5,3,2)$ であるから P_3 を実行できる。完了後，割り当て $(3,0,2)$ が返却され利用可能は $(8,3,4)$ となる。
3. P_1 の Need $(7,2,3) \le (8,3,4)$ であるから P_1 を実行できる。完了後，利用可能は $(8,4,4)$ となる。

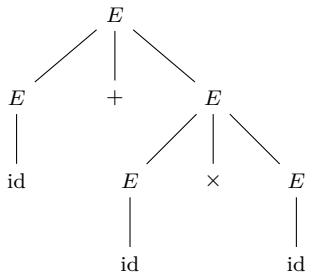
すべてのプロセスが完了できる実行順序が存在するので，この状態は**安全 (safe)** である。安全な実行順序の一例は

$$P_2 \to P_3 \to P_1.$$

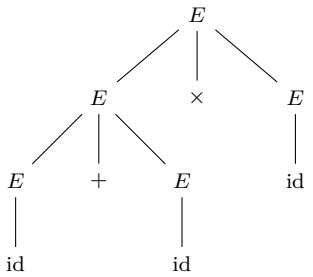
このように安全な実行順序の有無を調べる手法を**銀行家のアルゴリズム** (banker's algorithm) と呼ぶ。

4 【選択問題】計算理論

(1-1) 語 $\mathrm{id} + \mathrm{id} \times \mathrm{id}$ に対する構文木は以下の 2 通りが存在する。



木 1：+ を先に適用（× が下）



木 2：× を先に適用（+ が下）

木 1 は $E \to E + E$ を先に適用し，右側の E に $E \to E \times E$ を適用したものであり，$\mathrm{id} + (\mathrm{id} \times \mathrm{id})$ と解釈される。木 2 は $E \to E \times E$ を先に適用し，左側の E に $E \to E + E$ を適用したものであり，$(\mathrm{id} + \mathrm{id}) \times \mathrm{id}$ と解釈される。

同一の語に対して相異なる構文木が 2 つ存在するので，G_1 は曖昧である。

(1-2) 優先順位と結合則を文法の階層構造で表現する。変数を 3 段に分け，E（式，+ の階層），T（項，× の階層），F（因子，最も強く結合）とする。左結合とするため，各規則の左側を再帰させる（左再帰）。

$$G_2 = \big(\{E, T, F\},\ \{+, \times, (,), \mathrm{id}\},\ P_2,\ E\big),$$

$$P_2 = \{\ E \to E + T \mid T,\quad T \to T \times F \mid F,\quad F \to (E) \mid \mathrm{id}\ \}.$$

(1-3) 最左導出は以下のとおり。

$$E \Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow \mathrm{id} + T \Rightarrow \mathrm{id} + T \times F$$

$$\Rightarrow \mathrm{id} + F \times F \Rightarrow \mathrm{id} + \mathrm{id} \times F \Rightarrow \mathrm{id} + \mathrm{id} \times \mathrm{id}.$$

一意である理由：+ は必ず E の階層で，× は必ず T の階層でのみ生成される。T から + を含む式を直接導くことはできない（括弧を用いない限り）ため，$\mathrm{id} + \mathrm{id} \times \mathrm{id}$ を $(\mathrm{id} + \mathrm{id}) \times \mathrm{id}$ と解釈する構文木は存在しない。すなわち各語に対する構文木が階層により一意に定まる。

(2-1) 方針：$i = j$ の場合と $j = k$ の場合のどちらを検査するかを，計算の開始時に非決定的に選ぶ。

分岐 1（$i = j$ を検査する経路）：a を読むごとに X を積み，b を読むごとに X を取り去る。その後 c は自由に読み飛ばす。

分岐 2（$j = k$ を検査する経路）：a は自由に読み飛ばし，b で X を積み，c で X を取り去る。

状態を q_0（開始），q_1, q_2（分岐 1 用），q_3, q_4（分岐 2 用），q_f（受理）とすると，遷移は以下のとおり。

分岐の選択： $(\varepsilon, Z)/Z$ で $q_0 \to q_1$，および $(\varepsilon, Z)/Z$ で $q_0 \to q_3$

分岐 1：
q_1：$(a, Z)/XZ$，$(a, X)/XX$
$q_1 \to q_2$：$(b, X)/\varepsilon$
q_2：$(b, X)/\varepsilon$
$q_2 \to q_f$：$(\varepsilon, Z)/Z$
q_f：$(c, Z)/Z$
（$i = j = 0$ の場合に備え $q_1 \to q_f$：$(\varepsilon, Z)/Z$ も加える）

分岐 2：
q_3：$(a, Z)/Z$
q_3：$(b, Z)/XZ$，$(b, X)/XX$
$q_3 \to q_4$：$(c, X)/\varepsilon$
q_4：$(c, X)/\varepsilon$
$q_4 \to q_f$：$(\varepsilon, Z)/Z$
（$j = k = 0$ の場合に備え $q_3 \to q_f$：$(\varepsilon, Z)/Z$ も加える）

非決定性を用いている箇所：q_0 において，$i = j$ を検査する分岐 1 と $j = k$ を検査する分岐 2 のどちらへ進むかを，入力を読む前に非決定的に選択している点である。入力を見ただけではどちらの等式が成立するか事前に判定できないため，両方の可能性を並行して探索する必要がある。

(2-2) 語 $a^n b^n c^n$ は，$i = j$（$n = n$）を満たすと同時に $j = k$（$n = n$）も満たす。したがって小問 (2-1) のオートマトンでは，分岐 1 を選んだ計算でも分岐 2 を選んだ計算でも受理に至る。すなわちこの語には受理計算が 2 通り存在する。

文脈自由文法で L を生成しようとすると，同様に「a と b の個数を対応づける導出」と「b と c の個数を対応づける導出」の 2 系統が必要になり，$a^n b^n c^n$ はその両方から導出できてしまう。どのような文法を設計してもこの二重性を解消できないため，L は本質的に曖昧である。

(3) 文脈自由言語が補集合について閉じていると仮定する。

$L_1 = \{a^n b^n c^m \mid n, m \geq 0\}$ および $L_2 = \{a^m b^n c^n \mid n, m \geq 0\}$ はいずれも文脈自由言語である。

仮定より $\overline{L_1}$ と $\overline{L_2}$ はともに文脈自由言語である。文脈自由言語は和集合について閉じているから，$\overline{L_1} \cup \overline{L_2}$ も文脈自由言語である。再び仮定より，その補集合

$$\overline{\overline{L_1} \cup \overline{L_2}}$$

も文脈自由言語となる。

ところがド・モルガンの法則より

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$$

である。これは文脈自由言語でないことが知られている。矛盾。

したがって仮定が誤りであり，文脈自由言語は補集合について閉じていない。$\boxtimes$

6 【選択問題】電子回路と論理設計

(1-1) 初期状態では

$$A = 00000_2, \qquad Q = N = 1011_2.$$

(1-2) $D = 0011_2 = 3$ である。各回の推移は以下のとおり。

回	シフト直後の A	$A - D$ の符号	手順 3 後の A	手順 3 後の Q
1	00001	負（$1 - 3 < 0$）	00001（復元）	0110
2	00010	負（$2 - 3 < 0$）	00010（復元）	1100
3	00101	正（$5 - 3 \geq 0$）	00010	1001
4	00101	正（$5 - 3 \geq 0$）	00010	0011

（各回の手順 1 では A と Q を連結して左シフトする。例えば 1 回目は $A = 00000$，$Q = 1011$ から Q の最上位 1 が A に入って $A = 00001$，$Q = 0110$ となる。）

最終的に

$$Q = 0011_2 = \mathbf{3}\ (\text{商}), \qquad A = 00010_2 = \mathbf{2}\ (\text{剰余}).$$

$11 \div 3 = 3$ 余り 2 であるから一致する。

(1-3) 判定方法：減算を 2 の補数による加算 $A + (\overline{D} + 1)$ で実現すると，結果の最上位ビット（符号ビット）が 1 であれば負である。あるいは加算時の桁上げ出力 (carry out) が 0 であることでも判定できる。

復元の方法：$A - D$ が負であった場合，その結果に D を加算し直せばよい。すなわち加算器を用いて $A_{\text{復元}} = (A - D) + D$ を計算する。（減算前の A を別レジスタに保持しておいて書き戻す方法もあるが，加算器を再利用する方が回路規模が小さい。）

(2-1) 状態遷移図は以下のとおり。カウンタの値が 4 に達したことを表す信号を c とする。

状態	遷移	出力
S_{INIT}	無条件 $\to S_{\text{SFT}}$	$A \leftarrow 0$，$Q \leftarrow N$，カウンタ $\leftarrow 0$
S_{SFT}	無条件 $\to S_{\text{SUB}}$	`shift`$= 1$（A, Q を左シフト）
S_{SUB}	無条件 $\to S_{\text{FIX}}$	`sub`$= 1$（$A \leftarrow A - D$）
S_{FIX}	$c = 0 \to S_{\text{SFT}}$ $c = 1 \to S_{\text{DONE}}$	符号が正：$q_0 \leftarrow 1$ 符号が負：$A \leftarrow A + D$（復元），$q_0 \leftarrow 0$ カウンタ $\leftarrow$ カウンタ $+1$
S_{DONE}	停止（または S_{INIT} へ）	`done`$= 1$

(2-2) 状態割り当ては $S_{\text{SFT}} = (0,0)$，$S_{\text{SUB}} = (0,1)$，$S_{\text{FIX}} = (1,0)$，$S_{\text{DONE}} = (1,1)$ である。遷移表は次のとおり。

y_1	y_0	c	D_1	D_0
0	0	—	0	1
0	1	—	1	0
1	0	0	0	0
1	0	1	1	1

（$(y_1, y_0) = (1,1)$ は S_{DONE} であり，そこからの遷移は本問の対象外であるからドントケアとしてよい。）

$D_1 = 1$ となるのは $(y_1, y_0, c) = (0, 1, *)$ と $(1, 0, 1)$ である。$D_0 = 1$ となるのは $(0, 0, *)$ と $(1, 0, 1)$ である。ドントケアを活用して簡単化すると

$$D_1 = \overline{y_1}\, y_0 + y_1\, \overline{y_0}\, c, \qquad D_0 = \overline{y_1}\, \overline{y_0} + y_1\, \overline{y_0}\, c.$$

（D_0 は $\overline{y_0}(\overline{y_1} + c)$ とも書ける。）

(3-1)　S_{INIT} に 1 サイクル，繰り返しの 1 回あたり $S_{\mathrm{SFT}}, S_{\mathrm{SUB}}, S_{\mathrm{FIX}}$ の 3 サイクルを 4 回，最後に S_{DONE} に 1 サイクル要する。よって

$$1 + 3 \times 4 + 1 = \textbf{14 クロックサイクル}.$$

(3-2)　$D = 0$ の場合，手順 2 の $A - D$ は常に A そのものとなり負にならない。したがって毎回商のビットに 1 が入り，Q は最終的に 1111_2 となる。A には被除数の下位ビットが残るが，これは数学的に意味のある商・剰余ではない（実際には商が無限大に発散すべきところ，4 ビットで表せる最大値が出力されるだけである）。

追加すべき回路：除数レジスタ D の全ビットの NOR（D の全ビットが 0 のとき 1 を出力する回路）を設け，これを零除算エラー信号として制御回路に入力し，S_{INIT} から直ちにエラー状態へ遷移させる。

7 【選択問題】数学解析と信号処理

(1-1) 初期条件がすべて 0 であるから，各素子の電圧をラプラス変換すると，抵抗は $RI(s)$，インダクタは $sLI(s)$，コンデンサは $\dfrac{I(s)}{sC}$ となる。入力は大きさ E のステップ電圧であるからそのラプラス変換は $\dfrac{E}{s}$ である。

キルヒホッフの電圧則より

$$RI(s) + sLI(s) + \frac{I(s)}{sC} = \frac{E}{s}.$$

$I(s)$ について解くと

$$I(s) = \frac{E/s}{R + sL + \dfrac{1}{sC}} = \frac{E}{s\left(R + sL + \dfrac{1}{sC}\right)} = \frac{E}{sR + s^2L + \dfrac{1}{C}} = \frac{EC}{LCs^2 + RCs + 1}.$$

整理すると

$$I(s) = \frac{E}{L} \cdot \frac{1}{s^2 + \dfrac{R}{L}s + \dfrac{1}{LC}}.$$

(1-2) 与えられた値を代入すると $\dfrac{R}{L} = 3$，$\dfrac{1}{LC} = \dfrac{1}{1 \times 0.5} = 2$，$\dfrac{E}{L} = 6$ であるから

$$I(s) = \frac{6}{s^2 + 3s + 2} = \frac{6}{(s+1)(s+2)}.$$

部分分数分解すると

$$\frac{6}{(s+1)(s+2)} = \frac{6}{s+1} - \frac{6}{s+2}.$$

（$s = -1$ で $\dfrac{6}{-1+2} = 6$，$s = -2$ で $\dfrac{6}{-2+1} = -6$。）

逆ラプラス変換して

$$i(t) = 6\left(e^{-t} - e^{-2t}\right) \qquad (t \geq 0).$$

（検算：$i(0) = 0$ でありインダクタの電流が連続であることと整合する。また $t \to \infty$ で $i \to 0$ となり，コンデンサが充電されて直流を通さなくなることと整合する。）

(1-3) $I(s)$ の極は分母 $(s+1)(s+2) = 0$ の根であるから

$$s = -1, \qquad s = -2.$$

いずれも**実数**であり虚部を持たない。したがって応答は指数関数の和であり**振動的ではない**（過減衰, over-damped）。

一般に極は

$$s = \frac{-R/L \pm \sqrt{(R/L)^2 - 4/(LC)}}{2}$$

であり，根号の中身の符号で性質が決まる。振動的になるのは極が複素数となる場合，すなわち

$$\left(\frac{R}{L}\right)^2 - \frac{4}{LC} < 0 \quad \Longleftrightarrow \quad R < 2\sqrt{\frac{L}{C}}$$

のときである。したがって $R = 2\sqrt{L/C}$ を境に，これより R が小さくなると応答は振動的（不足減衰, under-damped）になる。

（本問では $2\sqrt{L/C} = 2\sqrt{1/0.5} = 2\sqrt{2} \approx 2.83 < 3 = R$ であるから，確かに非振動的である。）

(2-1)　$H(z)=\dfrac{1+z^{-1}}{1-0.5z^{-1}}$ の分子・分母に z を掛けると

$$H(z)=\frac{z+1}{z-0.5}.$$

よって

$$\text{極}: z=0.5, \qquad \text{零点}: z=-1.$$

極 $z=0.5$ は $|0.5|<1$ であり単位円の内側にある。因果的なシステムが BIBO 安定であるための条件はすべての極が単位円の内側にあることであるから，このシステムは **BIBO 安定**である。

(2-2)　$H(z)=\dfrac{Y(z)}{X(z)}$ より

$$Y(z)\left(1-0.5z^{-1}\right)=X(z)\left(1+z^{-1}\right).$$

逆 z 変換すると（z^{-1} は 1 サンプルの遅延に対応），

$$y[n]-0.5\,y[n-1]=x[n]+x[n-1],$$

すなわち

$$y[n]=0.5\,y[n-1]+x[n]+x[n-1].$$

インパルス応答：ヒントに従い

$$H(z)=\frac{1}{1-0.5z^{-1}}+\frac{z^{-1}}{1-0.5z^{-1}}$$

と分ける。第 1 項の逆変換は $0.5^n u[n]$，第 2 項は第 1 項を 1 サンプル遅延させたものであるから $0.5^{n-1}u[n-1]$。よって

$$h[n]=0.5^n u[n]+0.5^{n-1}u[n-1].$$

具体的には $h[0]=1$，$n\geq 1$ では

$$h[n]=0.5^n+0.5^{n-1}=0.5^{n-1}(0.5+1)=1.5\times 0.5^{n-1}.$$

（$h[1]=1.5$，$h[2]=0.75$，$h[3]=0.375$，……）

(2-3)　$z=e^{j\omega}$ を代入する。

$\omega=0$ のとき $z=1$ であるから

$$|H(e^{j0})|=\left|\frac{1+1}{1-0.5}\right|=\frac{2}{0.5}=\mathbf{4}.$$

$\omega=\pi$ のとき $z=-1$，すなわち $z^{-1}=-1$ であるから

$$|H(e^{j\pi})|=\left|\frac{1+(-1)}{1-0.5\times(-1)}\right|=\frac{0}{1.5}=\mathbf{0}.$$

低周波（$\omega=0$）で利得が最大の 4，高周波（$\omega=\pi$）で利得が 0 であるから，このシステムは**低域通過フィルタ**の特性をもつ。

(2-4)　$\omega=\pi$ は複素平面上の点 $z=e^{j\pi}=-1$ に対応する。このシステムの零点はちょうど $z=-1$ にあるため，周波数応答を評価する点が零点と一致する。伝達関数は零点で 0 となるので，$|H(e^{j\pi})|=0$ となる。すなわち単位円上に零点が存在する周波数では，その成分が完全に阻止される。