2022年度

慶應義塾大学入学試験問題

総合政策学部

数学または情報

注意事項1

1. 試験開始の合図があるまで、この問題冊子を開かないでください。
2. この冊子は全部で28ページです。
   - 数学の問題Ⅰ～Ⅵは3ページから10ページです。
   - 情報の問題Ⅰ～Ⅴは12ページから25ページです。
3. 試験開始の合図とともにすべてのページが揃っているか確認してください。
   ページの欠落・重複があった場合には、直ちに監督者に申し出てください。
4. 問題冊子の2ページに「注意事項2」があります。試験開始後必ず読んでください。
5. 数学・情報のいずれか1つを選択し、解答用紙の選択科目名の欄に科目名を記入し、
   選択科目マーク欄にマークしてください。
6. 問題冊子は、試験終了後必ず持ち帰ってください。
7. 受験番号と氏名は、解答用紙の所定の欄に必ず記入してください。
8. 解答用紙の「注意事項」を必ず読んでください。

情報Ⅰ

以下、法制度に関しては、日本のものについて考えるものとする。

(ア) 次の文章を読み、空欄 (1) ～ (5) にあてはまるものを選択肢から選び、その番号を解答欄にマークしなさい。

「海賊」とは、海上で略奪行為を働く盗賊だ。そこから転じた「海賊版」という言葉には、著作権を侵害した (1) は、コンテンツの「盗品」ととらえる発想がある。

２０２１年１月、改正著作権法が施行され、漫画、雑誌など著作物全般について海賊版のダウンロードが違法になった。この法改正までは大変な曲折があった。

発端は「漫画村」などいくつかのサイトによる海賊版被害が急拡大したこと。対策として１８年には政府の知的財産戦略本部で、ネット利用者の海賊版への接続を遮断する (2) の導入が議論された。だが遮断のためのアクセス検知により、憲法が保障する「 (3) 」が侵害されかねないとの猛反発に遭い、結論は見送られた。

文化庁が海賊版ダウンロードを違法化する範囲を従来の映像や音楽から静止画などにも拡大する方向性を打ち出した後も「ネット利用が萎縮する」などと反対論が噴出。１９年３月、自民党はいったんまとまった改正著作権法案の国会提出を断念した。その後、長編漫画の数コマ程度などの軽微なダウンロードは違法化対象から外すなど、ネット利用者の懸念に配慮することで、ようやく今回の改正法施行にこぎつけた。

（中略）

日本の著作権法はフランスやドイツなど大陸法系の流れをくむ。著作権を創作の動機を与える手段とみる英米法系の法思潮と違い、根幹にあるのは作者自身に帰属する「自然権」として正当化する発想だ。作者が氏名を表示する権利、作品を勝手に (4) されない権利などが重視されるのはそのためだ。

ただ、匿名のコンテンツも氾濫するネット時代に、「自然権だから著作権を守れ」というメッセージは利用者に響かないだろう。そこで権利者側の主張も「創造のサイクルのために必要」という (5) 的な考え方に振れつつある。創作者に利益が還元されないと次の創作につながらず、社会全体にとって不利益になるという考え方だ。しかしこちらも、二次創作が新たな創造の形として市民権を得るなか説得力は落ちている。

（出典：日本経済新聞 2021 年 2 月 21 日朝刊記事を一部改変）

【 (1) ～ (5) の選択肢】

(1) 改変　(2) アイデア　(3) 英米法　(4) ブロッキング　(5) 大陸法
(6) 法の下の平等　(7) 複製物　(8) 所有　(9) ミュート　(0) 通信の秘密

(イ) 産業財産権に関する説明として、正しいものを次の選択肢から１つ選び、その番号を解答欄 (6) にマークしなさい。

(1) 商標登録の出願では、消費者庁長官に願書を提出しなければならない。
(2) 商標権は、設定の登録の日から 20 年を超えて更新することができない。
(3) 特許権者は、発明者以外の者に対して特許発明の実施を許諾することはできない。
(4) 特許は、製品またはサービスとして実用化されていない段階の技術的思想について出願することはできない。
(5) 特許権は、特許出願前に外国において公然と知られていた発明については、取得することができない。

(ウ) 著作権法に関する説明として、正しいものを次の選択肢から１つ選び、その番号を解答欄 (7) にマークしなさい。

(1) 著作権法の目的は、産業の発達に寄与することである。
(2) 原著作物を翻案することにより創作した著作物を、共同著作物という。
(3) ネット配信用の動画コンテンツは、「映画の著作物」に該当しない。
(4) 事実の伝達にすぎない雑報および時事の報道は、言語の著作物に該当しない。
(5) 公表された著作物を、著作権者に無断で引用してはいけない。

(エ) 個人情報の保護に関する法律（個人情報保護法）に関する説明として、正しいものを次の選択肢から１つ選び、その番号を解答欄 (8) にマークしなさい。

(1) 個人情報取扱事業者は、あらかじめ本人の同意を得ないで、個人情報を取得してはならない。
(2) 個人情報取扱事業者は、個人情報を取り扱うに当たっては、その利用の目的をできる限り特定しなければならない。
(3) 個人情報取扱事業者は、本人の請求を受けた場合は、当該本人が識別される保有個人データを必ず消去しなければならない。
(4) 個人情報取扱事業者は、個人情報を取得する場合は、あらかじめ、その利用目的を、本人に通知しなければならない。
(5) 個人情報取扱事業者は、個人データの取扱いを委託することに伴って当該個人データを委託先に提供する場合、あらかじめ本人の同意を得なければならない。

情報 II

2 進法表現および 10 進法表現による固定小数点数と浮動小数点数の扱いと誤差について述べた次の文章の空欄 [(9)][(10)].[(11)][(12)][(13)]〜[(38)] に入るもっとも適切な数字を解答欄にマークしなさい。

ただし、浮動小数点数の場合、仮数部は 2 進法または 10 進法で表現され、基数部と指数部は 10 進法で表現されているものとする。

2 進法で、1.101_2 は、10 進法では、[(9)][(10)].[(11)][(12)][(13)]$_{10}$ となる。10 進法で、1.7_{10} は、2 進法では、小数第 4 位以下を切り捨てると、[(14)][(15)].[(16)][(17)][(18)]$_2$ となる。

次に、2 進法の浮動小数点数として、次の数を考える。

$$1.1111_2 \times 2^3$$

これは、10 進法の小数では、[(19)][(20)][(21)].[(22)][(23)]$_{10}$ となる。

また、同じ値を表す多くの表現があり、上記と同じ値を表す表現の例として下記のようなものがある。

$$11.111_2 \times 2^2$$

$$0.11111_2 \times 2^{\boxed{(24)}}$$

ここで、浮動小数点数の表現において、同じ値を表現しているものの中で小数点の左側が 1 以上基数未満の 1 桁になっているものを、正規化表現と呼ぶことにする。上記の例では、最初のものが正規化表現である。

計算機の処理では、仮数部や指数部は、一定の桁数に制限されることが多い。ここでは説明のために、小数第 3 位まで使えるものとし、指数部は、−63 から 64 の範囲を使えるものとする。また 10 進法を 2 進法に変換する場合には、正規化表現にした後で小数第 4 位以下は切り捨てるものとする。

10 進法で $2.2_{10} \times 10^1$ は、2 進法の正規化表現にされた浮動小数点数では、次のようになる。

$$\boxed{(25)}.\boxed{(26)}\boxed{(27)}\boxed{(28)}_2 \times 2^{\boxed{(29)}}$$

また、10 進法で $2.3_{10} \times 10^1$ は、仮数部の小数第 4 位以下が切り捨てられるため、$2.2_{10} \times 10^1$ と同じく、

$$\boxed{(25)}.\boxed{(26)}\boxed{(27)}\boxed{(28)}_2 \times 2^{\boxed{(29)}}$$

に変換されてしまう。

次に、二つの浮動小数点数の加算を考える。

$$1.110_2 \times 2^5 + 1.010_2 \times 2^4$$

指数部分が異なるとそのままでは仮数部同士を加算できないので、指数部分を大きな方に合わせると次のようになる。

$$1.110_2 \times 2^5 + 0.101_2 \times 2^5$$

仮数部を加算すると、

$$\boxed{(30)}\boxed{(31)}.\boxed{(32)}\boxed{(33)}\boxed{(34)}_2 \times 2^5$$

が得られる。

これを正規化表現にする場合、小数第 4 位以下が切り捨てられることに注意すると、

$$1.\boxed{(35)}\boxed{(36)}\boxed{(37)}_2 \times 2^{\boxed{(38)}}$$

が得られる。以上が、計算機内部での浮動小数点演算の基本的な処理手順である。また、最後の正規化表現にする部分で誤差が生じていることがわかる。

情報 III

順列とは、有限個の事象の重複のない並び順である。多くの実社会の問題は、最大価値を与える順列や、与えられた制約を満足する順列を求める問題に帰結できる。膨大な候補から正解や最適解を求めることや、そもそも正解が存在するかを人手で判断するのは、時間が掛かる上に間違えることもあるので、コンピュータで処理することが有効である。

(ア) 次の文章の空欄 (39)(40)(41)(42)(43)(44)〜(57)(58)(59)(60)(61)(62) には、あてはまるもっとも適切な数字を解答欄にマークしなさい。

ある工業製品の作業工程が A_1, A_2, A_3, A_4, A_5 の５工程あり、作業員が W_1, W_2, W_3, W_4, W_5 の５名いて、各作業員が１つの工程を担当し、作業工程順と担当作業員によって生産効率が変わるという状況で、作業工程順と担当作業員を決定せねばならないとしよう。

作業工程の順列は (39)(40)(41)(42)(43)(44) 通りあり、作業工程順と担当作業員のあらゆる候補数は (45)(46)(47)(48)(49)(50) 通りである。

もし作業工程 A_4 は必ず A_5 の直後という制約がある場合には、その条件を満たす作業工程順と担当作業員の候補数は (51)(52)(53)(54)(55)(56) 通り、作業工程 A_4 は必ず A_5 の直後という制約に加えて、作業工程 A_3 は必ず W_2 以外の作業員が行う制約がある場合には、この２つの制約を満足する作業工程順と担当作業員の候補数は (57)(58)(59)(60)(61)(62) 通りとなる。このようにもともとの検索範囲が膨大であっても、制約条件をプログラム中でうまく利用することで、検索範囲を狭め結果的に早く計算できる。

(イ) 次の文章の空欄 (63)(64)〜(87)(88) には、下の選択肢から最も適切なものを選び、その番号を解答欄にマークしなさい。

n 個の事象に対して順列を全て列記するプログラムの開発を考える。それぞれの事象は順列の中で一回しか現れないことを利用し、初期状態から２つの事象を入れ替える手続きを繰り返して、重複することなく順列を列記するアプローチを取る。事象の順列をプログラムで表すために、事象数と同数の変数 $H_1, \ldots, H_n$ を用いる。H は添え字１を最下位として、添え字が大きくなることを上位と表現する。この変数に事象を漏れなく、同じ順列を与えないように注意しながら代入することで順列を全て列記する。

例えば、３つの事象 A_1, A_2, A_3 がそれぞれ H_1, H_2, H_3 に代入されている順列を初期状態とした場合の、事象の入れ替えによる順列列記の例を図１に示す。図中の二重枠は前の状態から変化がない変数を表している。事象が n 個の時、同じ順列が現れないように工夫するなら、初期状態から (63)(64) 回事象の入れ替えを行い、初期状態の順列と、入れ替えを行う度にその時点の順列を出力することで、順列を全て列記できる。

$A_1, \ldots, A_{n-1}$ の $n-1$ 個の事象に対する順列を全て列記するための処理を処理 P_{n-1} とした時、事象 A_n を加えた n 個の事象の順列を全て列記する処理 P_n は、H_n に A_n を代入した状態で H_1 から H_{n-1} に対して処理 P_{n-1} を実施し、次に H_1 から H_{n-1} に代入されている事象から一つ選んで H_n の事象と入

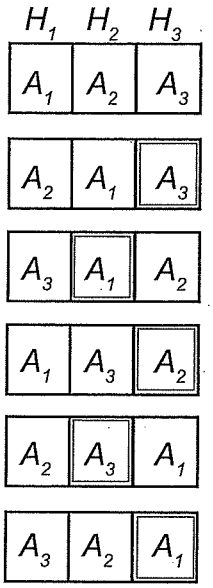


図１　順列列記の例

れ替えて処理 P_{n-1} を実施する、という手続きを繰り返すことに置き換えられる。H_n の事象入れ替えの際に、A_1 から A_n の事象が H_n に１回だけ代入されるようにすれば、同じ順列が現れることはない。つまり処理 P_n は H_n に代入される事象を入れ替えながら、処理 P_{n-1} を (65) (66) 回実施することと等価である。

処理 P_n において H_n に代入する事象の選択には複数の方法があるが、以下のアルゴリズムは計算効率が良い。

- n が奇数の場合には処理 P_{n-1} 実施後に H_n と H_1 の事象を入れ替える
- n が偶数の場合には最初の処理 P_{n-1} 実施後に H_1 と H_n の事象を入れ替え、次の処理 P_{n-1} 実施後には H_2 と H_n の事象を入れ替えというように H_1 から順に入れ替え場所を上位に移動する

事象数が n の場合、H_n の事象入れ替え操作は初期状態から (67) (68) 回行うことになる。

同様に処理 P_{n-1} を処理 P_{n-2} の繰り返しに置き換え、処理 P_{n-2} を処理 P_{n-3} の繰り返しに置き換え、というように、扱う事象数を減らした処理の繰り返しに順次置き換え、P_1 に至れば事象が１個しかないため、処理 P_0 は実施しない。

上記のアルゴリズムを適用して、$A_1, \ldots, A_n$ がそれぞれ $H_1, \ldots, H_n$ に代入されている初期状態から順列を全て列記する場合、H_n は、一定回数順列が出力される間同じ事象が留まり、それから次の事象へと順次変化する。図１はこのアルゴリズムを用いており、H_3 に配置される事象は２回連続で留まりながら、列記が進むにつれ A_3, A_2, A_1 と変遷している。

様々な n に対して、H_n に配置される事象が A_i の次に、A_i とは異なる事象 A_j となる変遷を $A_i \rightarrow A_j$ と表現するとして、上記のアルゴリズムを実施した場合に H_n に代入される事象の変遷に関する下表を完成させなさい。

総

事象数 (n)	H_n に代入される事象の変遷
2	$A_2 \to A_1$
3	$A_3 \to A_2 \to A_1$
4	$A_4 \to$ (69)(70) $\to$ (71)(72) $\to A_1$
⋮	⋮
8	$A_8 \to$ (73)(74) $\to$ (75)(76) $\to$ (77)(78) $\to$ (79)(80) $\to$ (81)(82) $\to$ (83)(84) $\to A_1$
⋮	⋮
11	$A_{11} \to$ (85)(86) $\to$ (87)(88) $\to \cdots$

【(63)(64)〜(87)(88) の選択肢】

(11) A_1　(12) A_2　(13) A_3　(14) A_4

(15) A_5　(16) A_6　(17) A_7　(18) A_8

(19) A_9　(20) A_{10}　(21) A_{11}　(22) n

(23) $n-1$　(24) $n+1$　(25) $n!$　(26) $n!-1$

(27) $(n-1)!$　(28) $n-i$　(29) $n+i$

情報 IV

4 ビットの 2 進数の加減算を行う回路を設計する手順を考える。ただし、$A+B$ は A と B の論理和（OR）を表し、$A \cdot B$ は A と B の論理積（AND）を表す。また、$\overline{A}$ は A の否定（NOT）を表す。$\overline{A+B}$ は A と B の論理和の結果を否定した否定論理和（NOR）であり、$\overline{A \cdot B}$ は A と B の論理積の結果を否定した否定論理積（NAND）である。

(ア) 次の文章の空欄 (89)(90)(91)(92)〜(101)(102)(103)(104)(105)(106) にあてはまる数字を解答欄にマークしなさい。

ある自然数に数値を加算してもとの自然数を 1 桁増やすことを考えよう。このとき加算する最小の数値を、ある基数を指定して、その基数の補数と呼ぶ。例えば、2 桁の 10 進数 46_{10} の 10（基数）の補数は、加算して 3 桁になる数なので 54_{10} となる。n ビットの 2 進数の場合も同様に、元の数を 2^n にするために補う数として、2 の補数を求めることができる。4 ビットの 2 進数を $A_3A_2A_1A_0$ とすると、

$$A_3A_2A_1A_0 + Y_3Y_2Y_1Y_0 = 10000_2$$

となる $Y_3Y_2Y_1Y_0$ は、$A_3A_2A_1A_0$ の 2 の補数となる。具体的な手順は次のようになる。

手順 1　もとの 2 進数の各ビットを反転（1 であれば 0 に、0 であれば 1 に変換）する。
手順 2　反転して得られた 2 進数に 1 を加算する。

この 2 の補数を用いて整数を表現する方法を考える。4 ビットの 2 進数を次のように正の整数、ゼロ、負の整数と対応させることにしよう。

10 進数	2 進数	10 進数	2 進数
−8	1000	0	0000
−7	1001	1	0001
−6	1010	2	0010
−5	1011	3	0011
−4	1100	4	0100
−3	1101	5	0101
−2	1110	6	0110
−1	1111	7	0111

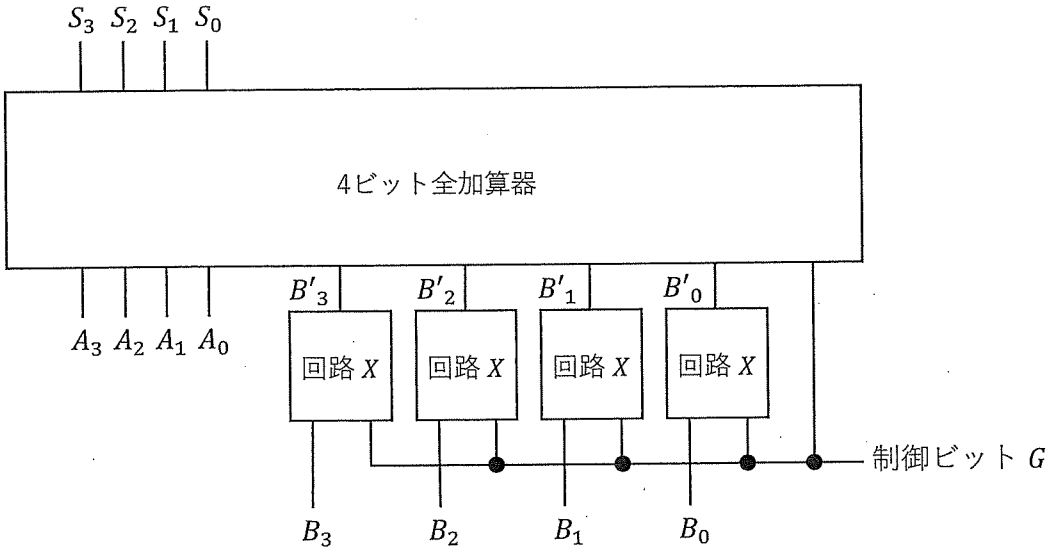
このとき、10 進数 6_{10} に対応する 2 進数 0110_2 の 2 の補数は (89)(90)(91)(92) $_2$ であり、10 進数の (93)(94) $_{10}$ に相当する。このような正負の数値の割り当て方法が 2 の補数を用いた整数表現である。ただし、4 ビットの 2 進数 0000_2 の 2 の補数は 0000_2 とする。同様に 2 の補数を用いた 16 ビットの整数表現を用いると、(95)(96)(97)(98)(99)(100) $_{10}$ 〜 (101)(102)(103)(104)(105)(106) $_{10}$ までの整数を扱うことができる。

(イ) 次の文章の空欄 (107) にあてはまるもっとも適したものを下の選択肢から選び、その番号を解答欄にマークしなさい。

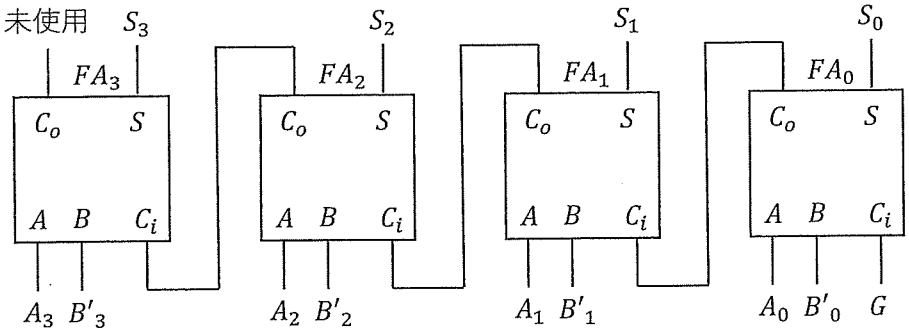
2の補数を用いた4ビットの整数表現を用いると、次のように加算器を用いて4ビットの2進数の減算を行うことが可能となる。ただし、計算結果が −8〜7 の範囲に収まる整数を対象とするものとする。

$$A_3A_2A_1A_0 - B_3B_2B_1B_0 = A_3A_2A_1A_0 + (-B_3B_2B_1B_0)$$
$$= A_3A_2A_1A_0 + (B_3B_2B_1B_0\text{の 2 の補数})$$

4ビットの2進数演算 $A_3A_2A_1A_0 \pm B_3B_2B_1B_0 = S_3S_2S_1S_0$ を対象に、次図の上のような回路構成を考えよう。多くの CPU では、図のような回路で実際に加減算を行う。4ビット全加算器回路は、次図の下のように4つの全加算器から構成される。各全加算器（FA）は A、B、および下位ビットからの桁上がり C_i を入力とし、加算した結果を和 S および桁上がり C_o として出力する。



4ビット加減算器の全体構成



4ビット全加算器の内部構成

上図の回路では、制御ビット $G = 0$ の場合に加算 $A_3A_2A_1A_0 + B_3B_2B_1B_0 = S_3S_2S_1S_0$ が実行され、$G = 1$ の場合に減算 $A_3A_2A_1A_0 - B_3B_2B_1B_0 = S_3S_2S_1S_0$ が実行されることとする。入力 B_k と G か

ら B'_k を出力する回路を X とすると、回路 X の真理値表は次のようになる。

B_k	G	B'_k
0	0	0
0	1	1
1	0	1
1	1	0

真理値表より、B'_k を求める論理式は次のようになる。

$$B'_k = \boxed{(107)} \tag{1}$$

【(107) の選択肢】

(1) $\overline{B_k} \cdot \overline{G}$　(2) $\overline{B_k} \cdot G$　(3) $B_k \cdot \overline{G}$　(4) $B_k \cdot G$
(5) $\overline{B_k} \cdot \overline{G} + \overline{B_k} \cdot G$　(6) $\overline{B_k} \cdot \overline{G} + B_k \cdot \overline{G}$　(7) $\overline{B_k} \cdot \overline{G} + B_k \cdot G$　(8) $\overline{B_k} \cdot G + B_k \cdot \overline{G}$
(9) $\overline{B_k} \cdot G + B_k \cdot G$　(0) $B_k \cdot \overline{G} + B_k \cdot G$

(ウ) 次の文章の空欄 (108)(109) にあてはまる数字を解答欄にマークしなさい。

回路 X を NAND（否定論理積）回路のみから構成することを考える。式（1）に対して、復元の法則とド・モルガンの定理を順に適用すると、式（2）のように変形できる。

$$B'_k = \overline{\overline{\overline{B_k} \cdot G} \cdot \overline{B_k \cdot \overline{G}}} \tag{2}$$

式（2）はさらに変形が可能であり、1つの回路 X を構成する NAND 回路の数は、最小で (108)(109) 個となる。論理式の変形には、次に示す論理演算の諸定理を用いてよい。

総

公理	恒等の法則
$1 + A = 1$ $0 \cdot A = 0$	$0 + A = A$ $1 \cdot A = A$
同一の法則	**補元の法則**
$A + A = A$ $A \cdot A = A$	$A + \overline{A} = 1$ $A \cdot \overline{A} = 0$
交換の法則	**結合の法則**
$A + B = B + A$ $A \cdot B = B \cdot A$	$A + (B + C) = (A + B) + C$ $A \cdot (B \cdot C) = (A \cdot B) \cdot C$
分配の法則	**吸収の法則**
$A \cdot (B + C) = A \cdot B + A \cdot C$ $(A + B) \cdot (A + C) = A + B \cdot C$	$A \cdot (A + B) = A$ $A + A \cdot B = A$ $A + \overline{A} \cdot B = A + B$ $\overline{A} + A \cdot B = \overline{A} + B$
復元の法則	**ド・モルガンの定理**
$\overline{\overline{A}} = A$	$\overline{A \cdot B} = \overline{A} + \overline{B}$ $\overline{A + B} = \overline{A} \cdot \overline{B}$

情報 V

ソートとは、長さ 2 以上の有限数列が与えられた時に、その項をある順序に従って並べ替えた数列を作ることとする。ただし、問題を簡単にするため、与えられた数列の各項はすべて異なるものとし、並べ替える順序は小さい順だけを考える。例えば、数列 $7, 1, 4, 2$ にソートを行った結果は $1, 2, 4, 7$ となる。

(ア) 次の文章の空欄 (110)〜(115) に入るもっとも適切な数字を解答欄にマークしなさい。

ソートのアルゴリズムにはいろいろなものがあるが、ここではバブルソートについて考える。バブルソートとは次のような考え方によるソートである。

1. まず数列の第 1 項と第 2 項を比較し、それが小さい順になっていれば（第 1 項が第 2 項より小さければ）何もせず、そうでなければ第 1 項と第 2 項を交換する。
2. 同様に第 2 項と第 3 項、第 3 項と第 4 項、…に対して、それが小さい順になっていれば何もせず、そうでなければその 2 つの項を交換する。
3. 数列の最後の 2 つの項まで上の手順を行うと、最後の項が数列の中の最大の数になるので、最後の項を除いた数列に対して再びアルゴリズムを適用してソートを行えばよい。

これをアルゴリズムの形で書くと次のようになる。

アルゴリズム 1:

```
変数 n の値を与えられた数列の長さ、変数 a_1, ..., a_n の値を与えられた数列の各項とする。
変数 i の値を最初 n − 1 とし、1 ずつ減らしながら 1 まで処理 A を繰り返す。
処理 A の始め
    変数 j の値を最初 1 とし、1 ずつ増やしながら i まで処理 B を繰り返す。
    処理 B の始め
        もし a_j > a_{j+1} なら処理 C を実行する。
        処理 C の始め
            a_j の値と a_{j+1} の値を交換する。
        処理 C の終わり
    処理 B の終わり
処理 A の終わり
a_1, ..., a_n の値を出力する。
```

アルゴリズム 1 において、長さ 4 の数列が与えられた時の処理 C の実行回数の最小値は (110)、最大値は (111) である。一般に長さ n の数列が与えられた時の処理 C の実行回数の最大値は $\dfrac{(112)}{(113)}n^2 - \dfrac{(114)}{(115)}n$

である。

(イ) 次の文章の空欄 (116)(117)〜(118)(119) に入るもっとも適切な数字を解答欄にマークしなさい。

アルゴリズム 1 では、与えられた数列によっては実行の途中ですべて小さい順に並んでしまうことがある。その場合は残りの処理は必要がないので、その時点でアルゴリズムを終了させることで効率を良くすることができる。アルゴリズム 1 をそのように変更すると次のようになる。なお、左端の数字は、変更箇所を示すための行番号である。4 行目、10 行目、13 行目が新しく挿入された行である。

アルゴリズム 2:

1: 変数 n の値を与えられた数列の長さ、変数 $a_1, \ldots, a_n$ の値を与えられた数列の各項とする。
2: 変数 i の値を最初 $n-1$ とし、1 ずつ減らしながら 1 まで処理 A を繰り返す。
3: 処理 A の始め
4: 　変数 f の値を 0 とする。(命令 D)
5: 　変数 j の値を最初 1 とし、1 ずつ増やしながら i まで処理 B を繰り返す。
6: 　処理 B の始め
7: 　　もし $a_j > a_{j+1}$ なら処理 C を実行する。
8: 　　処理 C の始め
9: 　　a_j の値と a_{j+1} の値を交換する。
10: 　　変数 f の値を 1 とする。(命令 E)
11: 　　処理 C の終わり
12: 　処理 B の終わり
13: 　もし $f = 0$ なら、$a_1, \ldots, a_n$ の値を出力して、アルゴリズムを終了する。(命令 F)
14: 処理 A の終わり
15: $a_1, \ldots, a_n$ の値を出力する。

1, 2, 3, 4 という 4 つの項を並べ替えた順列は全部で 24 個あるが、それぞれをアルゴリズム 1 とアルゴリズム 2 でソートした時の処理 B の実行回数を比較する。実行回数が変わらない順列は (116)(117) 個ある。また、実行回数の差の最大値は (118)(119) である。

(ウ) 次の文章の空欄 (120)〜(121) に入るもっとも適切なものを下の選択肢から選び、その番号を解答欄にマークしなさい。

アルゴリズム 1 をアルゴリズム 2 に変更する際に、誤って次のように変更してしまったとする。それぞれの誤りを含むアルゴリズムの動作は、正しいアルゴリズム 2 と比べてどのように変化するかを考える。

- 命令Dを、4行目ではなく、1行目と2行目の間に挿入した場合、(120)
- 命令Eを書き忘れた場合、(121)

【(120)～(121)の選択肢】

(1) 常に正しい結果が出るが、実行時間が長くなる場合がある。
(2) 与えられた数列が最初から小さい順に並んでいる場合を除いて、誤った結果が出る。
(3) 処理Aを1回実行した時点で小さい順に並んでいる場合を除いて、誤った結果が出る。
(4) 処理Bを1回実行した時点で小さい順に並んでいる場合を除いて、誤った結果が出る。
(5) 与えられた数列が最初から小さい順に並んでいる場合を除いて、アルゴリズムが終了しなくなる。
(6) 処理Aを1回実行した時点で小さい順に並んでいる場合を除いて、アルゴリズムが終了しなくなる。
(7) 処理Bを1回実行した時点で小さい順に並んでいる場合を除いて、アルゴリズムが終了しなくなる。