令和２年度

千葉大学先進科学プログラム入学者選考課題

論述課題

情報

（9:00–12:00）

注意事項

1. この冊子は，監督者から解答を始めるよう合図があるまで開いてはいけません。
2. 問題冊子に印刷または製本の不具合がある場合は，手を上げて申し出てください。
3. 問題すべてに解答してください。
4. 解答用紙は，課題ごとに解答用紙を分けて使用してください。解答用紙は何枚使用しても構いません。全ての解答用紙に受験番号を必ず記入してください。
5. 検査室に用意してある資料は自由に使用してかまいません。ただし，諸君が持参した教科書，参考書，ノート，パソコンなどの使用は禁止します。
6. 携帯電話やスマートフォン等の電子機器はすべて電源を切り，カバンにしまってください。
7. その他，監督者の指示に従ってください。

[I]

次に示すプログラムに関し，次ページの問いに答えよ。

```
#include <stdio.h>

int
main(void)
{
    int a, b, c, d, e, f;
    int n = 2809;

    f = 100;
    while (f < n)
        f *= 100;
    a = n;
    c = 0;
    e = 0;
    while (f > 1) {
        f /= 100;
        b = a / f;
        d = 1;
        c = c * 10 + d;
        while (c * d <= b) {
            d++;
            c++;
        }
        a -= --c * --d * f;
        c += d;
        e = e * 10 + d;
    }
    printf("%d\n", e);
    printf("%d\n", a);
    return 0;
}
```

問 1 このプログラムを実行した際に，表示される 2 つの値は何と何になるか示しなさい。

問 2 (07) 行目の 2809 を 2808 に変更した場合，表示される 2 つの値は何と何になるか示しなさい。

問 3 (07) 行目で変数に代入する値を n，(28) 行目で表示される値を e，(29) 行目で表示される値を a としたとき，これらの値の関係を簡潔な数式で表しなさい。

問 4 (09) 行目の 100 を 64 に変更したとき，プログラムが変更前のプログラムと同じ動作をするためには，他にどこを変更しなければならないか。

(xx) 行目の yyy を zzz に変更

という形式で，すべての変更点を示しなさい。

[II]

千葉県を 12 の地域に分け, 隣り合う地域が異なる色となるように塗り分けることを考える。

問 1 まず簡単のために図 1 に示す地図 A を考える。地図 A の 4 つの地域にそれぞれに整数を割り振る。このとき, 地図 A では 1 つの地域が隣接する地域が最大 3 つであることから, 各地域の隣接関係 `areaA` は下のコードのように表記できる。ここで, `-1` は終端（それ以上隣接する地域が存在しない）を意味する。同様にして, 図 2 に示す地図 B について, 各地域の隣接関係 `areaB` を記述しなさい。

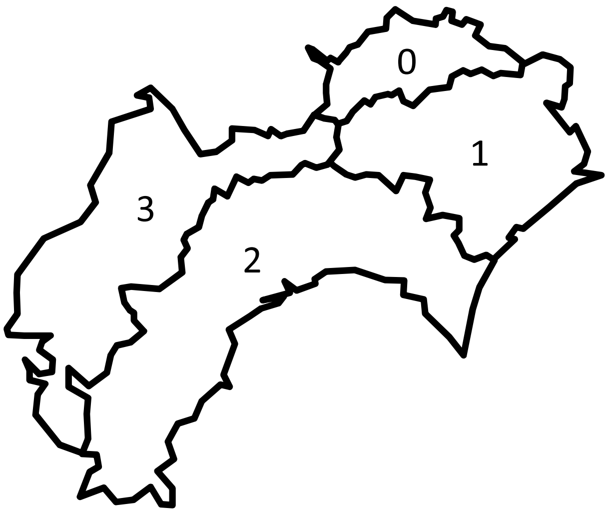

図 1: 地図 A

```
int areaA[4][4] = {
  {1, 3, -1},
  {0, 2, 3, -1},
  {1, 3, -1},
  {0, 1, 2, -1}
};
```

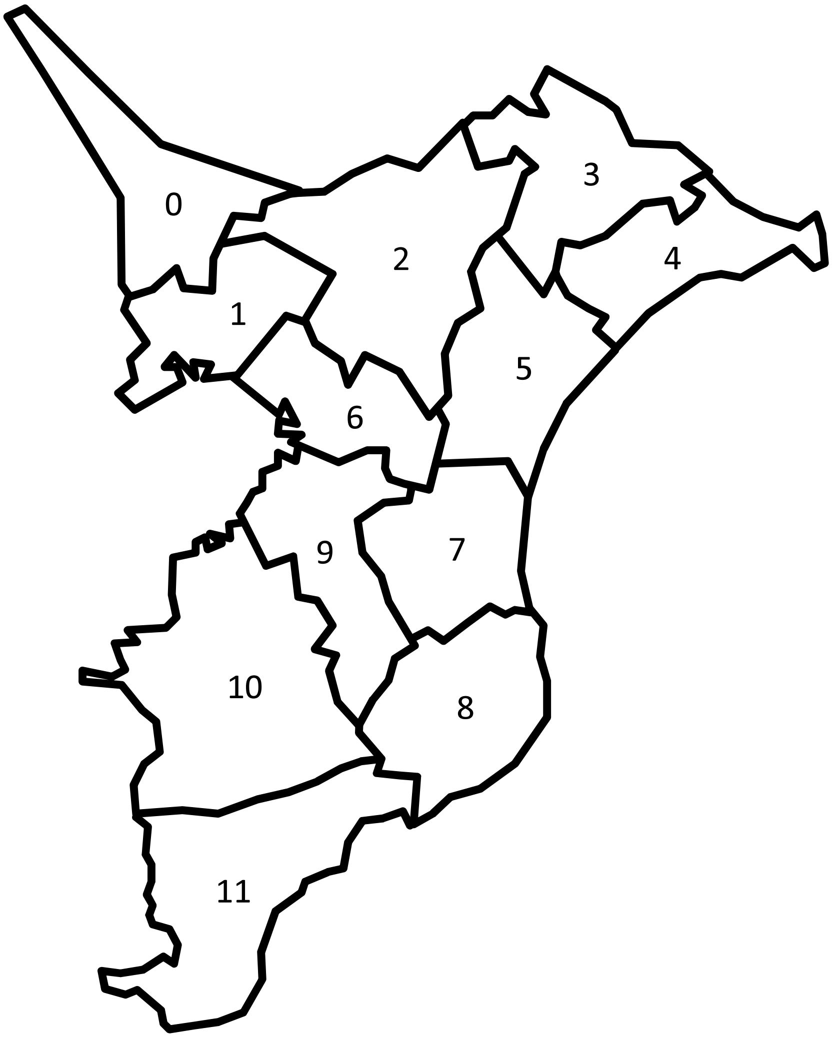

図 2: 地図 B

問 2 問 1 で定義した areaB を用い, 地域を表す整数 2 つを引数として与えると, 2 つの地域が隣接していれば 1 を, 隣接していなければ 0 を返す関数 isbeside を記述しなさい。

問 3 図 2 の地図 B は, 4 色あれば塗り分けられる。ここでは各地域に色を表す番号 0∼3 を割り当てる。問 1 の隣接関係 areaB, 問 2 の関数 isbeside を使って, 各地域に割り当てる色の番号を配列 color に格納するコードを記述しなさい。

[III]

行列とは，値を式 (1) のように縦横に並べたものである。値の 1 つ 1 つを成分と言い，$a_{i,j}$ と表す。ここで i や j は，縦や横に何番目の成分かを表す。

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \tag{1}$$

2 つの行列 A，B の積 AB は

$$AB = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,n} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n,1} & b_{n,2} & \cdots & b_{n,n} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,n} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,n} \end{pmatrix} \tag{2}$$

と定義される。ここで

$$c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j} \tag{3}$$

である。例えば $c_{2,2}$ の計算には、以下に灰色で示す成分が必要である。

$$AB = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,n} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n,1} & b_{n,2} & \cdots & b_{n,n} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,n} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,n} \end{pmatrix} \tag{4}$$

問 1　行列 A，B，C を二次元配列で表現し，式 (2) で表される行列の積を行うコードを以下のように実装する。ここで，A[i][j] は $a_{i+1,j+1}$ を表す。

```
/*
 * 行列の積を求めるコード
 */

double A[n][n];
double B[n][n];
double C[n][n] = {0.0};

/* A,B は初期化されているとする */

int i, j, k;
for( i=0; i<n; i++){
  for( j=0; j<n; j++){
    /* C[i][j] を計算する */

    [    空   欄    ]

  }
}
```

(a) 上の ［ 空欄 ］ に当てはまる部分を記述し，コードを完成しなさい。記述は複数行にわたっても構わない。

(b) 実装した行列の積のコードを実行したとき，1 つの $c_{i,j}$ を計算するのに行なった乗算の回数，および $c_{1,1}$ から $c_{n,n}$ までの全ての $c_{i,j}$ を計算するのに行なった乗算の回数を n を用いて表しなさい。

問 2　問 1 で実装したコードの，簡略化された計算機上での動作について考える。

簡略化された計算機は演算器，キャッシュメモリ，メインメモリから構成される (図 1)。四則演算などの処理を行う演算器は，キャッシュメモリに存在する行列の成分にのみアクセスできる。演算に必要な成分がキャッシュメモリに存在しない場合は，メインメモリから 1 つずつキャッシュメモリに読み込む。キャッシュメモリに同時に格納できる成分の数は限られており，空きがなければ既に存在するいずれかの成分の 1 つを選択し上書きする。上書きされて失われた成分が必要な場合は，再度メインメモリから読み込まねばならない。

この計算機では 1 回の乗算にかかる時間は T_c，メインメモリからキャッシュメモリへの 1 つの成分の読み込みにかかる時間は T_m である。乗算と読み込みは同時には行えない。演算器がキャッシュメモリにアクセスする時間と加算にかかる時間は無視できる。問 2 では，A と C は常にキャッシュメモリに存在する。キャッシュメモリには B 全体も格納できるが，計算開始時には B はキャッシュメモリに存在しない。キャッシュメモリからメインメモリへの書き込み時間は無視できる。

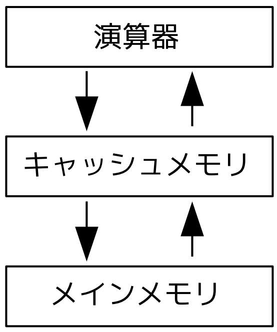

図 1: 簡略化された計算機

(a) C を計算するには，メインメモリからキャッシュメモリへ B の成分を何回読み込む必要があるか。その回数を n を用いて表しなさい。

(b) C を計算するのに必要な時間 T を，n，T_c，T_m を用いて表しなさい。

(c) $\dfrac{T_m}{T_c}$ と n の関係に注目し，時間 T が n のサイズによってどのように変化するか議論しなさい。

問 3　キャッシュメモリに同時にすべての成分を格納できない場合を想定し，以下の問いに答えなさい。

(a) キャッシュメモリに B の成分を n 個だけ格納できるとする。問 1 のコードのループをうまく変更すると，メインメモリから B の成分を読み込む回数を，キャッシュメモリに B 全体を格納できる場合と同じにできる。そのコードを記述しなさい。

(b) 行列が大きく，A と C が常にキャッシュメモリ上に存在できない場合を考える。C を計算するのに，メインメモリから成分を読み込む回数を最小化するようなアルゴリズムを考案しなさい。ヒント: 縦と横の両方向に，成分を少しずつ読み込むと良い。