令和 6 年度

千葉大学先進科学プログラム入学者選考課題

課題論述

情報

(12:00 – 15:00)

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かまいません。すべ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。

問題

1

以下の C 言語プログラムに関する問いに答えなさい。

問 1　方程式 $ax^2 + bx + c = 0$ の a，b，c を入力とし，その解 x を出力するプログラムを作成したい。次のプログラムの空欄（ア）～（セ）を適切に埋め，それらを解答用紙に書きなさい。

```
#include <stdio.h>
#include <math.h>

int main(void)
{
  double a, b, c;
  double root, root1, root2;
  double discriminant, realPart, imaginaryPart;

  printf("方程式 ax^2 + bx + c = 0 の解を求めます。¥n");

  printf("係数 a を入力してください。: ");
  scanf("%lf", &a);

  printf("係数 b を入力してください。: ");
  scanf("%lf", &b);

  printf("係数 c を入力してください。: ");
  scanf("%lf", &c);

  (ア) {
       (イ) {
            (ウ) {
                printf("無限個の解が存在します。¥n");
            } else {
                printf(" (エ) ¥n");
            }
       } else {
            root = -c / b;
            printf("1 次方程式の解: %lf¥n", root);
       }
  } else {
        discriminant = b * b - 4 * a * c;

         /* 次ページに続く */
```

```
            (オ)        {
                root1 =          (カ)          ;
                root2 =          (キ)          ;
                printf("実数解 1: %lf¥n", root1);
                printf("実数解 2: %lf¥n", root2);
            }          (ク)          {
                root =    (ケ)    ;
                printf("重解: %lf¥n", root);
            }          (コ)          {
                realPart =    (サ)    ;
                imaginaryPart =          (シ)          ;
                printf("虚数解 1:    (ス)    ¥n", realPart,imaginaryPart);
                printf("虚数解 2:    (セ)    ¥n", realPart,imaginaryPart);
            }
        }

        return 0;
}
```

問2

（1）下記プログラムを実行したとき何が出力されるか説明しなさい。

```
#include <stdio.h>
int bisection (int arr[], int left, int right, int target)
{
    int mid;
    while (left <= right) {
        mid = left + (right - left) / 2;

        if (arr[mid] == target) {
            return mid;
        } else if (arr[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
        return -1;
}

int main(void)
{
    int arr[] = {2, 4, 6, 8, 10, 12, 14, 16, 18, 20};
    int size = sizeof(arr) / sizeof(arr[0]); /*配列 arr[]の要素数*/
    int target, result;

    printf("値を入力してください: ");
    scanf("%d", &target);

    result = bisection (arr, 0, size - 1, target);
    printf("%d ¥n", result);
    return 0;
}
```

問 2

（2） 問 2-1 のプログラム中の bisection 関数を下記表現に変更するとき空欄（ア）～（エ）を埋め，それらを解答用紙に書きなさい。

```
int bisection(int arr[], int left, int right, int target)
{
    int mid;
    if (left > right) {
        return    (ア)    ;
    }

    mid = left + (right - left) / 2;

    if (arr[mid] == target) {
        return    (イ)    ;
    } else if (arr[mid] < target) {
        return bisection (        (ウ)        );
    } else {
        return bisection (        (エ)        );
    }
}
```

問 2

（3） 問 2（1），（2）と同様のアルゴリズムを用いて方程式 $f(x)=0$ において，$a<x<b$ の範囲に一つの実数解があるとき，その解を求めたい。ただし，方程式を数値的に解く場合，x を厳密に求めることは難しく近似解で良いこととする。具体的には，$|f(\hat{x})| < \text{tolerance}$ を満足する $\hat{x}$ を「方程式の近似解」と呼ぶ。このとき，tolerance は十分小さい値をとる。また，$f(x)$ は連続関数であることを仮定する。

下記はそのプログラムである。このとき空欄(ア)～(キ)を埋め，それらを解答用紙に書きなさい。ただし、空欄(A)には $f(x)$，例えば x*x-2, cos(x)-0.3 などが入る。

```
#include <stdio.h>
#include <math.h>

double f(double x){
    return(   (A)   );
}

double binarySearch(double a, double b, double tolerance)
{
    double mid = (a + b)/2;

    if (fabs(   (ア)   ) < tolerance) {      /*fabs(y)は y の絶対値*/
            return   (イ)   ;
    } else if (   (ウ)   <0) {
            return binarySearch(      (エ)      );
    } else {
            return binarySearch(      (オ)      );
    }
}/* 次ページに続く */
```

```
int main(void)
{
    double a, b;
    double tolerance = 0.0001;

    printf("探索範囲の最小値を入力してください。: ");
    scanf("%lf", &a);
    printf("探索範囲の最大値を入力してください。: ");
    scanf("%lf", &b);

    if (   (カ)   > 0){

        printf("探索範囲が不適切です。解を導出できません。¥n");
    } else {
        printf("方程式の（近似）解は x = %.6f です。¥n",
        binarySearch(    (キ)    ));
    }
    return 0;
}
```

2

以下の C 言語プログラムに関する問いに答えなさい。

問１　以下は関数 f を再帰的に呼び出して，ある値を求めるプログラムである。

（１）以下のプログラムを実行するとプログラム開始から終了までに関数 f が何回呼び出されるか答えなさい。

（２）以下のプログラムは何を求めるプログラムであるか答えなさい。

```
#include <stdio.h>
#include <stdlib.h>

#define A 403
#define B 221

int f(int a, int b)
{
	if(a%b==0)
		return(b);
	return(f(b, a%b));
}



int main(void)
{
	printf("%d¥n", f(A,B));
	return 0;
}
```

問 2　以下は，整数をソート（整列）するプログラムである。各行の先頭の（）内の数字は行番号を表している。以下の問いに答えなさい。

（1） このプログラムを実行すると、main 関数から sort 関数が引数(0, 9)で呼ばれ，呼ばれた sort 関数の中で次に引数(0, 4)で sort 関数を呼んでいる。 引数の列挙は以下のようになる。

(0, 9), (0, 4), …

3 番目以降を補って呼ばれるときの引数を最初からすべて列挙しなさい。ただし，処理は単一 CPU で実行され，プログラムに書かれている順に逐次的に実行されると仮定する。

（2）このプログラムの(22)行目から(45)行目の for ループの動作について説明しなさい。

```
(01) #include <stdio.h>
(02) #include <stdlib.h>
(03) #define MAX 10
(04) int a[MAX]={5,7,14,9,25,
(05)   11,8,3,16,6};
(06) void sort(int i, int j)
(07) {
(08)   int m=(i+j)/2;
(09)   int p,q,n;
(10)   int b[MAX];
(11)   int c[MAX];
(12)   if(i==j) return;
(13)   sort(i,m);
(14)   sort(m+1,j);
(15)   for(p=0;p<m-i+1;p++)
(16)          b[p]=a[i+p];
(17)   /*前半の結果を配列ｂに格納*/
(18)   for(q=0;q<j-m;q++)
(19)          c[q]=a[m+1+q];
(20)   /*後半の結果を配列ｃに格納*/
(21)   p=0;q=0;
(22)   for(n=0;n<j-i+1;n++)
(23)   {
(24)          if(q>=j-m)
(25)          {
(26)                 a[n+i]=b[p];
(27)                 p++;
(28)          }
(29) /* 右上に続く */
```

```
(30)          else if(p>=m-i+1)
(31)          {
(32)                 a[n+i]=c[q];
(33)                 q++;
(34)          }
(35)          else if(b[p]<c[q])
(36)          {
(37)                 a[n+i]=b[p];
(38)                 p++;
(39)          }
(40)          else
(41)          {
(42)          a[n+i]=c[q];
(43)          q++;
(44)          }
(45)   }
(46) }
(47) int main(void)
(48) {
(49)   int i;
(50)   sort(0,MAX-1);
(51)
(52)   for(i=0;i<MAX;i++)
(53)   {
(54)          printf("%d ",a[i]);
(55)   }
(56)   printf("¥n");
(57)   return 0;
(58) }
```

問 3　レーベンシュタイン距離は文字列の類似度を測る指標の一つである。文字列 a と文字列 b のレーベンシュタイン距離は文字列 a を文字列 b に一致させるために文字列 a に対して行う(i)1 文字挿入　(ii)1 文字削除　(iii)1 文字置換の回数の合計の最小値で与えられる。この距離はスペルミスをした単語から正しいスペルの単語を推定するときなどに用いられる。以下のプログラムはレーベンシュタイン距離を求めるプログラムである。(一部出題のために空欄にしている。) 以下の問いに答えなさい。

（1）“man” と “women” のレーベンシュタイン距離を求めなさい。

（2）関数 m3 の動作を説明しなさい。

（3）関数 levensthein では文字列 a の最後尾へ 1 文字を挿入した場合，最後尾の 1 文字を削除した場合，最後尾の 1 文字を置換した場合の 3 通りの場合について操作の回数を計算し，それらを比較してレーベンシュタイン距離を求めている。関数 levensthein 内では最後尾に行った操作の部分を除いた部分のレーベンシュタイン距離を求めるために

① levensthein(sa-1,sb)

② levensthein(sa-1,sb-1)

③ levensthein(sa,sb-1)

の 3 回，関数 levensthein を再帰的に呼び出している。①～③それぞれに対して「挿入」,「削除」,「置換」のいずれかであるか答えなさい。

（4）関数 levensthein の戻り値では levensthein(sa-1,sb-1)の再帰呼び出しの結果に変数 cost の値を加えている．この変数 cost の取りうる値とその意味を答えなさい。

（5）以下のプログラム中の空欄 [（ア）], [（イ）]を埋めなさい。

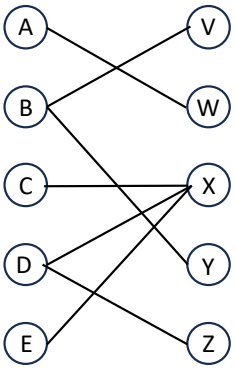
```
#include <stdio.h>
#include <string.h>

char a[]="man";
char b[]="women";

int m3(int i, int j, int k)
{
        int m=i;
        if(j<m) m=j;
        if(k<m) m=k;
        return(m);
}

/* 次ページに続く */
```

```
int levensthein(int sa, int sb)
{
	int cost;
	if(sa==0) return( (ア) );
	if(sb==0) return( (イ) );
	if(a[sa-1]==b[sb-1])
		cost=0;
	else
		cost=1;
	return(m3(
		levensthein(sa-1,sb)+1,
		levensthein(sa-1,sb-1)+cost,
		levensthein(sa,sb-1)+1
	));
}

int main(void)
{
	int sa=strlen(a);
	int sb=strlen(b);

	printf("%d¥n",levensthein(sa,sb));

	return 0;

}
```

3

ある高校のテニスクラブで，大会に出場するため，男女混合ダブルスの一対一のペアを作ることになった。以下の問いに答えなさい。

問１　5 人の男子 A，B，C，D，E が，5 人の女子 V，W，X，Y，Z とペアを組む。男子と女子，それぞれにペアになってもよいと思う相手を指名してもらった結果，下の左図のように描けた。線で結ばれているのは，双方がペアになってもよいということを表している。これを 2 次元配列として，下の右図で表すことにする。1 はペア可能，0 はペア不可能を表している。



	V	W	X	Y	Z
A	0	1	0	0	0
B	1	0	0	1	0
C	0	0	1	0	0
D	0	0	1	0	1
E	0	0	1	0	0

このとき，できるだけ多くのペアを作りたい。上の例では，一例として，A-W，B-V，C-X，D-Z のペアが考えられ，最大のペア数は 4 である。

与えられた 2 次元配列に対して，最大ペア数とそのときのペアの一例を出力する C 言語のプログラムを次のように作成した。空欄（ア）と（イ）を埋めなさい。なお，bool 型は，真であれば true あるいは 1 を返し，偽であれば false あるいは 0 を返す論理型である。

```
#include <stdio.h>
#include <stdbool.h>

#define M 5  /* 男子の数 */
#define N 5  /* 女子の数 */

bool matching(bool matrix[M][N], int u, bool wariate[N], int pair[N])
{
    for (int v = 0; v < N; v++) {
        if (matrix[u][v] && !wariate[v]) {
            wariate[v] = true;
            if (    (ア)     || matching(          (イ)          )){
                pair[v] = u;
                return true;
            }
        }
    }
    return false;
}

int maxPair(bool matrix[M][N])
{
    int pair[N];
    for (int i = 0; i < N; i++) {
        pair[i] = -1;
    } /* 女子 i に対するペアの初期化 */

    int result = 0;
    for (int u = 0; u < M; u++) {
        bool wariate[N];
        for (int i = 0; i < N; i++) {
            wariate[i] = false;
        } /* 男子 u に対する女子の割当の有無の初期化 */

        if (matching(matrix, u, wariate, pair)) {
            result++;
        } /* 男子 u に対するペアが見つかれば，ペア数を増やす．  */
    }

    printf("最大ペア数は，%d です．¥n", result);
    printf("そのときのペアの一例は，次のとおりです:¥n");
    for (int i = 0; i < N; i++) {
        if (pair[i] != -1) {
            printf("男子 %d と女子 %d がペアになる．¥n", pair[i], i);
        }
    }

    return result;
}  /* 次ページに続く */
```

```
int main(void)
{
    bool matrix[M][N] = {{0, 1, 0, 0, 0},
                         {1, 0, 0, 1, 0},
                         {0, 0, 1, 0, 0},
                         {0, 0, 1, 0, 1},
                         {0, 0, 1, 0, 0}};
    maxPair(matrix);
    return 0;
}
```

問２　問１では，最大数のペアを求めたが，ペアが成立しない男女も出てしまう。そこで，各自が希望の順位を示すこととする。簡単のため，３人の男子 A，B，C と３人の女子 X，Y，Z の場合で考える。例えば，この合計６名に，それぞれ，次のような希望があるとする。

男子の希望：

	A	B	C
第１希望	Z	Z	X
第２希望	X	Y	Z
第３希望	Y	X	Y

女子の希望：

	X	Y	Z
第１希望	A	C	C
第２希望	B	A	B
第３希望	C	B	A

このとき，３人の男子と３人の女子を一対一で組み合わせるので，全部で次の６通りの組合せがある。

組合せ１

男子	女子
A	X
B	Y
C	Z

組合せ２

男子	女子
A	X
B	Z
C	Y

組合せ３

男子	女子
A	Y
B	X
C	Z

組合せ４

男子	女子
A	Y
B	Z
C	X

組合せ５

男子	女子
A	Z
B	X
C	Y

組合せ６

男子	女子
A	Z
B	Y
C	X

組合せ２では，C は，第３希望の Y とペアになっており，第１，２希望の X，Z とペアではない。一方，Z は，第２希望の B とペアになっており，第１希望の C ではない。このため，組合せ２は，C と Z の両者がともに合意できる，より希望に近いペアが存在するため，両者に不満が残り，不安定な組合せと言える。

一方，組合せ１は，男子と女子の両者がともに合意できる，より希望に近いペアが存在しないため，安定な組合せと言える。例えば，A は，第２希望の X とペアであるが，第１希望の Z は第１希望の C とペアになっており，A が不満に思っても，Z は第１希望の男子とペアになっているので，両者がともに合意できる，より希望に近いペアが存在しない。組合せ１については，他の５名も同様であり，安定な組合せと言える。

以上を踏まえて，残りの組合せ３，４，５，６について，それぞれ安定な組合せか不安定な組合せか述べなさい。特に，不安定な組合せの場合は，上の組合せ２の「C と Z」のように，より希望に近いペアを全て具体的に書きなさい。

問３　問２のように，男子と女子の希望が任意に与えられた場合，安定な組合せを１つ見つけ出すＣ言語のプログラムを次のように書いた。空欄（ウ）～（カ）を埋めなさい。

```
#include <stdio.h>
#include <stdbool.h>

#define N 4 /* 男子と女子のそれぞれの数 */

/* 女子 w が，男子 m よりも m1 のほうが希望順位が上であれば，true を返す関数.  */
bool wmyorimom1(int wPrefer[N][N], int w, int m, int m1)
{
       for (int i = 0; i < N; i++){
              if (wPrefer[w][i] == m1) return true;
              if (wPrefer[w][i] == m) return false;
       }
}

void stablePair(int mPrefer[N][N], int wPrefer[N][N])
{
       int wPair[N];
       bool mFree[N];

       for (int i = 0; i < N; i++) {/* 初期化 */
              wPair[i] = -1; /* 女子のペアの初期化 */
              mFree[i] = false; /* 男子のペアの初期化 */
       }
       int freeman = N;

       while (freeman > 0){ /* ペアになっていない男性がいる限り，続ける.  */
              int m;
              for (m = 0; m < N; m++)
                     if (mFree[m] == false) break;

              for (int i = 0; i < N && mFree[m] == false; i++){
                     int w = mPrefer[m][i];

                     if (wPair[w] == -1){
                            wPair[w] = m;
                            mFree[m] = true;
                            freeman--;
                     }
                     else{ /* 女子 w のペアがいるならば，*/
                            int m1 = wPair[w];
                            if (                （ウ）                ){
                                          （エ）
                                          （オ）
                                          （カ）
                            }
                     }/* else 文の最後 */
              }/* for 文の最後 */
       }/* while 文の最後 */            /* 次ページに続く */
```

```
        printf("安定な組合せ：¥n");
        for (int i = 0; i < N; i++)
                printf(" 男子%dが，女子%dとペア．¥n", wPartner[i],i+N);
} /* stablePair関数の最後 */

int main(void)
{
    int mPrefer[N][N] = /* 男子A,B,C,Dの希望順位 */
    {
        {1, 0, 2, 3}, /* 男子Aの女子W,X,Y,Zに対する希望順位 */
        {0, 3, 1, 2}, /* 男子Bの女子W,X,Y,Zに対する希望順位 */
        {1, 0, 2, 3}, /* 男子Cの女子W,X,Y,Zに対する希望順位 */
        {3, 2, 1, 0}  /* 男子Dの女子W,X,Y,Zに対する希望順位 */
    };
    int wPrefer[N][N] = /* 女子W,X,Y,Zの希望順位 */
    {
        {2, 3, 1, 0}, /* 女子Wの男子A,B,C,Dに対する希望順位 */
        {3, 1, 0, 2}, /* 女子Xの男子A,B,C,Dに対する希望順位 */
        {0, 2, 1, 3}, /* 女子Yの男子A,B,C,Dに対する希望順位 */
        {2, 1, 0, 3}  /* 女子Zの男子A,B,C,Dに対する希望順位 */
    };

    stablePair(mPrefer,wPrefer);

    return 0;
}
```

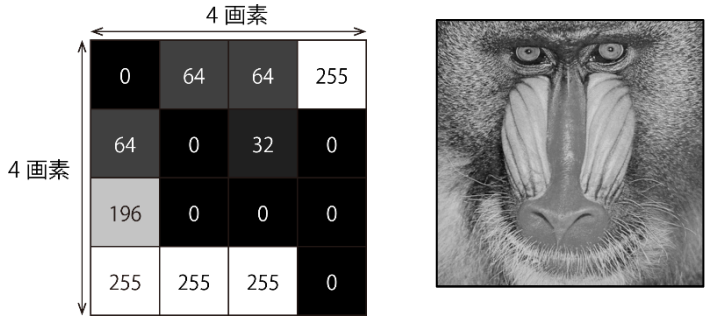
問4　問3のプログラムの出力を求めなさい。

4

Python のプログラムに関する次の記述を読んで設問に答えなさい。なお，解答の際にはモジュールのインポートは行わないこと。

デジタル画像とはキャンバス上に縦横の格子状に並んだ画素に値を記録し，その値に応じた色に置き換えることで表された画像のことをいう。画素値は 0 以上 255 以下の整数値をとるものとし，画素の色と値との関係は以下の通りとする。このような画像をグレースケールの画像を呼ぶ。

例えば，図１（a）に示すグレースケールのデジタル画像において，このデジタル画像のキャンバスの幅と高さはそれぞれ 4 画素であり，様々な画素値の画素が配置されている。なお，各画素の中央には便宜的に画素値を表示してあり，さらに各画素の境界を示す格子模様を付加してある。



(a) 4 画像×4 画素　　(b) 512 画素×512 画素

図１　デジタル画像の例

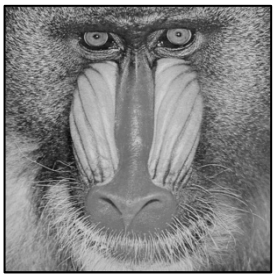
図 1(a)に示すデジタル画像は 2 次元リスト `canvas1` を用いてコード 1 のように表現される。

```
canvas1 = [
    [ 0,   64,  64, 255],
    [ 64,   0,  32,   0],
    [196,   0,   0,   0],
    [255, 255, 255,  0],
]
```

コード 1

デジタル画像はキャンバスの幅，高さを増やしていくにつれてより精細な表現ができるようになり，デジタルカメラやイメージスキャナなどの装置を利用して高精細な映像を記録することができる。一例として，幅５１２画素，高さ５１２画素のデジタル画像を図 1(b)に示す。

問 1　次のコード 2 には与えられた画像の画素値を 0.5 倍して返却する関数 darker を示している。具体的には，引数として任意の幅 width 画素，高さ height 画素のキャンバス canvas が与えられたとき，全ての画素の画素値を 0.5 倍し，さらに整数型に変換して返却する。例えば図 2(a)に示す画像に対してこの関数を実行すると，図 2(b)のように全体が暗くなったように見える。




(a) 元の画像　　(b) 処理後の画像

図 2

```
def darker(canvas, width, height):

  for h in range(height):
    for w in range(width):
      # 画素値を 0.5 倍し，数値型を int に変換して格納する
      canvas[h][w] = int(0.5 * canvas[h][w])
  return canvas
```

コード 2

このことを参考にして，図 3 の入力画像にコード 3 に示す invert 関数を実行する。

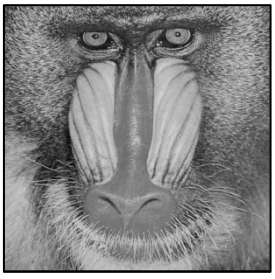
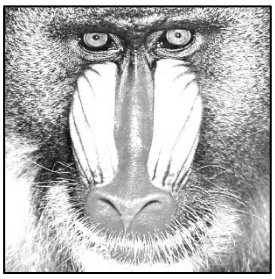
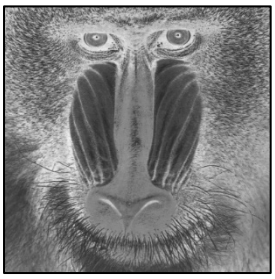


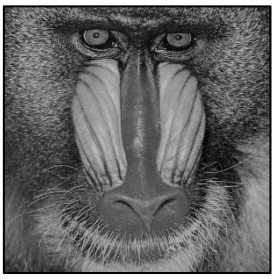
図 3 入力画像

```
def invert(canvas, width, height):

  for h in range(height):
    for w in range(width):
      canvas[h][w] = 255 - canvas[h][w]
  return canvas
```

コード 3

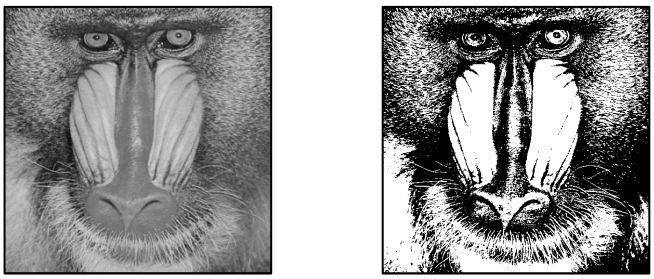
このときの結果としてもっとも適切なものを次のア～エから 1 つ選び答えなさい。



ア　　イ　　ウ　　エ

問 2　画素の色を黒（画素値 0）から白（画素値 255）まで 256 段階で表現する画像をグレースケール画像と呼ぶのに対し，画素の色が黒（画素値 0），または白（画素値 255）しか取り得ない画像のことを二値画像と呼ぶ。いま，入力として与えられたグレースケール画像に対して，各画素の画素値がある整数の閾値 th 未満の場合には黒，th 以上のときはに白に置き換えて，二値画像を作成する。例えば図 4(a)に示す入力画像に対して閾値 th=96 としてこの処理を実行すると，図 4(b)のようになる。このような二値画像を返却する関数 binarize をコード 4 のよう作成する。引数として任意の幅 width 画素，高さ height 画素のキャンバス canvas，および閾値 th が与えられたとき，**空欄 1** を記述しなさい。



(a) 入力画像　　（b） 二値画像（th=96）

図 4

```
def binarize(canvas, width, height, th):

  for h in range(height):
    for w in range(width):

  空欄 1

  return canvas
```

コード 4

問 3　グレースケール画像において，画像を画素値の集まりと捉えることで，データの代表値から画像の性質を推し量ることができる。

(1)　入力されたデジタル画像の画素値の平均値を返却する関数 average をコード 5 のように作成する。引数として任意の幅 width 画素，高さ height 画素のキャンバス canvas が与えられたとき，**空欄 2** を記述しなさい。

```
def average(canvas, width, height):

  #平均値を格納する変数を 0 で初期化
  ave = 0
```

空欄 2

```
  # 平均値を返却
  return ave
```

コード 5

(2)　入力されたデジタル画像の画素値の分散を返却する関数 variance をコード 6 のように作成する。引数として任意の幅 width 画素，高さ height 画素のキャンバス canvas が与えられたとき，**空欄 3** を記述しなさい。必要に応じて前問(1)の関数 average を利用して構わない。なお，分散とはデータの散らばり度合いを表す指標であり，n個のデータの値 $x_1, x_2, \cdots, x_n$の平均値を$\bar{x}$とするとき，分散s^2は

$$s^2 = \frac{1}{n}\{(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + \cdots (x_n - \bar{x})^2\}$$

と表される。

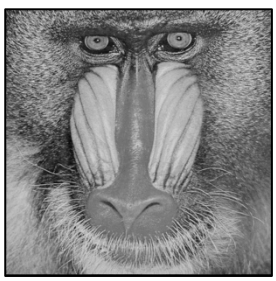
```
def variance(canvas, width, height):

  #分散を格納する変数を 0 で初期化
  var = 0
```

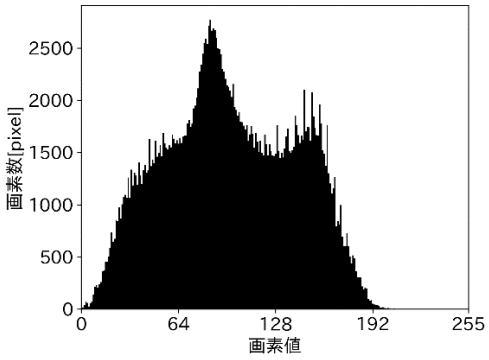
空欄 3

```
  # 分散を返却
  return var
```

コード 6

(3)　図 5 (a)に示すデジタル画像に対して図 5(b)には横軸に画素値を，縦軸にそれぞれの画素値の度数をとったヒストグラムを示す。画像のヒストグラムによってその画像の濃淡の分布を一目で確認することができる。ヒストグラムを表示するための度数分布を配列として返却する関数 fdt をコード 7 のように作成する。引数として任意の幅 `width` 画素，高さ `height` 画素のキャンバス `canvas` が与えられたとき，**空欄 4** を記述しなさい。ただし，度数分布における画素値の階級幅は 1, 階級は 0 以上 255 以下の 256 区間とし，関数 `fdt` は要素数 256 の 1 次元リストを返却し，その i 番目の要素には階級値 i の度数が格納されているものとする。





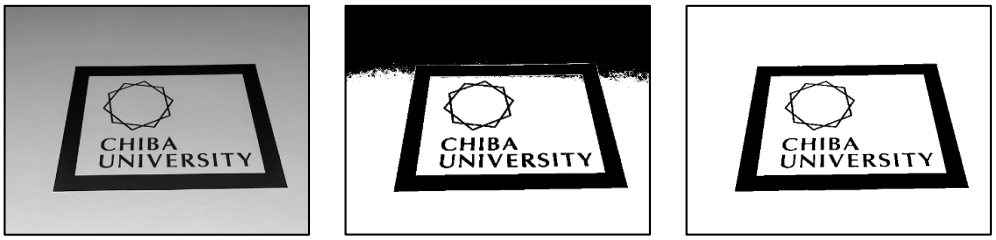
(a) デジタル画像　　(b)画像のヒストグラム

図 5

```
def fdt(canvas, width, height):

  #度数分布を格納するリストを初期化
  hist = [[0] * 256]

  空欄 4

  # 度数分布のリストを返却
  return hist
```

コード 7

問 4　白い紙に印字された図形を撮影して得られたグレースケール画像を図 6 (a)に示す。この画像から二値画像に変換して，印字された領域（前景）とそれ以外の領域（背景）とに二分することを考える。例えば th=128 のときの二値画像（図 6 (b)）では，画像の上部が実際には印字が無いにもかかわらず黒くなってしまっている。従って，適切な閾値 th を設定することが重要である。



(a)グレースケール画像　(b)二値画像（th=128）　(c) 二値画像（th=80）

図 6

このような画像に対する適切な閾値はつぎのようにして求めることができる。まず，閾値 th 未満の画素値の画素数と平均値をそれぞれn_1，m_1とおき，さらに th 以上の画素値の画素数と平均値をそれぞれn_2，m_2とおく。このとき$D = n_1 n_2 (m_1 - m_2)^2$を最大にする閾値を求めればよい。例えば，図 6(a)の画像の場合の最適な閾値は th=80 であり，このときの二値画像を図 6(c)に示す。印字された領域だけが黒くなっていることがわかる。

(1)　まず, ある閾値 th における D を計算する関数 calculate_D をコード 8 のように作成する。引数として任意の幅 width 画素，高さ height 画素のキャンバス canvas および閾値 th が与えられたとき，**空欄 5** を記述しなさい。なお，n_1ないしn_2が 0 になるような閾値は不適とし，-1 を返却するようにすること。

(2)　つぎに，閾値 th の取り得る全ての値についてそれぞれ D を計算して，D を最大化する最適な閾値を求める関数 calculate_threshold をコード 9 のように作成する。引数として任意の幅 width 画素，高さ height 画素のキャンバス canvas が与えられたとき，**空欄 6** を記述しなさい。なお，前問で作成した関数を用いても構わない。

```
def calculate_D(canvas, width, height, th):
  # 画素数と平均値を格納する変数を 0 で初期化
  m1, n1, m2, n2 = 0, 0, 0, 0
```

空欄 5

```
  # D を計算
  D = n1*n2*((m1 - m2)**2)

  # D を返却
  return D
```

コード 8

```
def calculate_threshold(canvas, width, height):
  # D の最大値を格納する変数を 0 で初期化
  max_D = 0.0
  # D を最大化する閾値 opt_th を 0 で初期化
  opt_th = 0

  # D を最大化する閾値を探索
  for th in range(0, 256):
```

空欄 6

```
  return opt_th
```

コード 9