次の問8は必須問題です。必ず解答してください。

問8　次のプログラムの説明及びプログラムを読んで，設問に答えよ。

〔プログラムの説明〕

64（8×8）画素からなる表示領域がある。この表示領域中の一つの画素を指定して，その画素を含む同じ色の領域を，指定した色で塗り替えるプログラムである。

ある一つの画素について，その画素の上下左右の4方向に隣接した画素の中に同じ色のものがあれば，それらの画素は同じ色の領域内にあるものと判定する。このようにして領域内にあると判定された隣接する各画素について，同様の判定を繰り返し，指定した色で塗り替えていく。

(1) 大きさ10×10の2次元配列Image（各添字の範囲は0～9）を用意する。各画素の色は，配列Imageの一部（各添字の範囲は1～8）に保持する。

(2) 色は，黒（■），灰（▒），白（□）の3色で，それぞれを値1，2，3で表す。

(3) 塗り替えたい領域中の開始点を示す一つの画素を，変数VSとHSで指定する。VSとHSは，その画素に対応する配列Imageの要素の縦と横方向の添字である。

(4) 塗り替えたい色を，変数NCで指定する。

(5) プログラムは，Image[VS，HS]から現在の画素の色を取得し，その画素を含む同じ色の領域を，色NCで塗り替える。

(6) 大域変数Image，NC，VS，HSには，正しい値が設定されているものとする。

(7) 配列VPos，HPosの添字は，1から始まる。

(8) 図1は，VS＝5，HS＝3として，Image[5，3]を塗り替えたい領域内の開始点に指定し，NC＝1として，塗り替える色を黒（■）に指定した場合の，プログラムの実行例である。

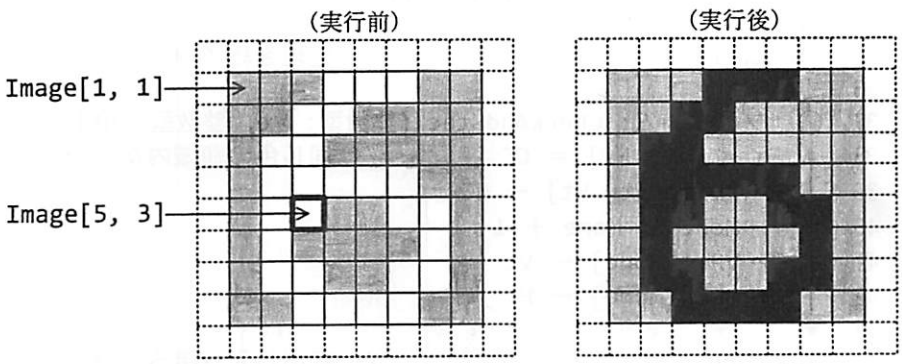


図1　プログラムの実行例（NC＝1，VS＝5，HS＝3の場合）

〔プログラム〕

```
(行番号)
 1  ○符号なし8ビット整数型: Image[10, 10]      /* 画素の色情報 */
 2  ○整数型: VS, HS                          /* 開始点はImage[VS,HS] */
 3  ○符号なし8ビット整数型: CC, NC           /* 色CCの領域を色NCで塗替え */
 4  ○整数型: More                            /* 処理待ちの画素の数 */
 5  ○整数型: VPos[64], HPos[64]              /* 処理待ちの画素の位置情報 */
 6
 7  ○プログラム: Main
 8  ○整数型: V, H                        /* 縦(V)と横(H)方向の添字用 */
 9  ○符号なし8ビット整数型: Wall         /* 表示領域の外周に格納する値 */
10
11  ・CC ← Image[VS, HS]                 /* 開始点の現在の色を取得 */
12  ▲ CC = NC
13  |   ・Return                         /* 処理を終了 */
14  ▼
15  ・ Wall ← 0
16  ■ V: 1, V ≦ 8, 1                     /* Vを1～8として外周に値を設定 */
17  |   ・Image[V, 0] ← Wall
18  |   ・Image[V, 9] ← Wall
19  ■
20  ■ H: 1, H ≦ 8, 1                     /* Hを1～8として外周に値を設定 */
21  |   ・Image[0, H] ← Wall
22  |   ・Image[9, H] ← Wall
23  ■
24  ・More ← 0
25  ・CheckAndStack(VS, HS)   /* 開始点を処理待ちの画素として登録 */
26  ■ More > 0                /* More>0の間，次の処理を繰り返す。 */
27  |   ・V ← VPos[More]
28  |   ・H ← HPos[More]
29  |   ・More ← More − 1
30  |   ・CheckAndStack(V − 1, H)
31  |   ・CheckAndStack(V, H − 1)
32  |   ・CheckAndStack(V + 1, H)
33  |   ・CheckAndStack(V, H + 1)
34  ■
35  ・Return                             /* 処理を終了 */
36
37  ○副プログラム: CheckAndStack(整数型: Vt, 整数型: Ht)
38  ▲ Image[Vt, Ht] = CC                 /* 同じ色の領域内か？ */
39  |   ・Image[Vt, Ht] ← NC
40  |   ・More ← More + 1
41  |   ・VPos[More] ← Vt
42  |   ・HPos[More] ← Ht
43  ▼
44  ・Return                             /* 呼出し元へ戻る。 */
```

設問　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

このプログラムに関する先輩技術者（以下，先輩という）と新人技術者（以下，新人という）の会話である。

先輩：プログラムの動作を理解するには，処理の流れを追跡してみることが大切です。まず，画素の色がどのような順序で塗り替えられていくか，図1のデータが与えられたものとして，最初の五つが塗り替えられる順序を1，2，…，5で示してみてください。

新人：はい。追跡してみます。結果は，［　a　］のようになりました。

先輩：そうですね。では，プログラムを検討していきましょう。まず，元の画素数は8×8ですが，配列Imageの大きさは10×10で，行番号15～23で最外周の配列要素に値0を設定しています。これは何のための処理でしょうか。

新人：はい，［　b　］からです。しかし，表示領域の要素数64に対して，32の配列要素に値を設定するのは，随分無駄な処理のように思えます。

先輩：では，一般にm×n画素からなる表示領域を考えたとき，行番号15～23で値を設定する要素数は，どういう式で表せますか。

新人：はい，［　c　］となります。ということは，表示領域の画素数m×nが大きくなるほど，この要素数は相対的に小さくなっていくのですね。

先輩：次に，最外周の配列要素に設定する値について考えてみましょう。行番号15では，変数Wallに値0を設定しています。与えられた仕様では，画素の色を表す値は1～3の範囲だからこれでよいのですが，もしも，符号なし8ビットで表せる値0～255がすべて色の指定に使えるとした場合，行番号15をどう変更したらよいですか。例を一つ挙げてみてください。

新人：行番号15を［　d　］と変更するのも一つの方法です。

先輩：次は，行番号12～14について考えてみます。これは何の処理ですか。

新人：現在の色と同じ色で塗り替えようとしているなら，以降の処理をせずに呼出し元に戻ります。塗り替えても，結局元どおりなので無駄ですから。

先輩：それも理由の一つではあるけれど。では，このプログラムから行番号12～14を取り去ってみましょう。そして，図2の①～③に示す，1画素，2画素，3画素からなる三つの白色の領域の例について，同じ色で塗り替えようとし

た場合のプログラムの動作を追跡してみてください。変数 VS，HS には，太枠で示した画素を指定するものとします。

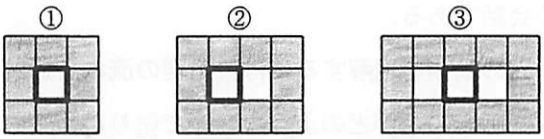


図2　1画素，2画素，3画素からなる領域

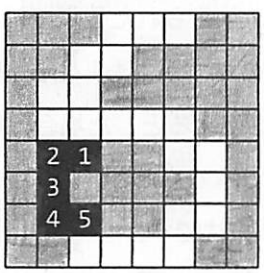
新人：では，追跡してみます。結果は，次の表のとおりです。この結果から，行番号12～14は省略できない必要な処理であることが分かりました。

表　追跡の結果

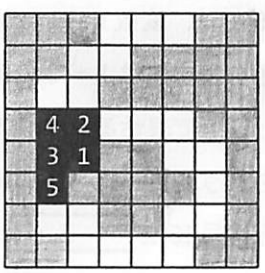
テストケース	プログラムの動作
図2の①（1画素からなる領域）	e
図2の②（2画素からなる領域）	f
図2の③（3画素からなる領域）	変数 More の値が増加していき配列の添字の上限を超える。

aに関する解答群

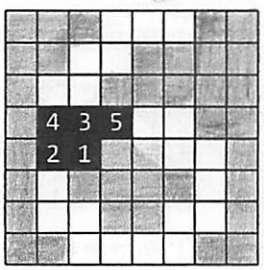
ア



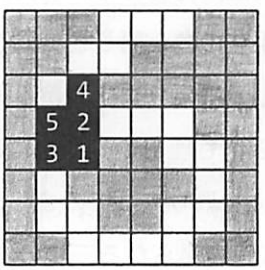
イ



ウ



エ



bに関する解答群

ア　これらの要素が参照されることはないが，添字から表示領域内かどうかが分かる

イ　これらの要素も色 NC での塗替えの対象とすることで，処理を簡素化できる

ウ　配列 Image の各添字の範囲チェックを省略できる

エ　配列 VPos と HPos の各添字の範囲チェックを省略できる

cに関する解答群

ア　$2(m+1)(n+1)$　　イ　$2(m+n)$

ウ　$2(m+n+2)$　　エ　$2(m+n-2)$

dに関する解答群

ア　・Wall ← CC　　イ　・Wall ← NC

ウ　・Wall ← 256 − CC　　エ　・Wall ← 255 − NC

e, fに関する解答群

ア　仕様どおりに同じ色で塗り替えて正しく終了する。

イ　塗替えは一度も実行せずに終了する。

ウ　塗り替えるべき領域に含まれない周辺の画素まで塗り替えて終了する。

エ　変数 More の値が増加していき配列の添字の上限を超える。

オ　変数 More の値は一定値以下であるが処理が無限にループする。