1．下に示したC言語のプログラムにおいて，行(X)から書かれているforループでは，配列aに0から3までの整数の乱数が1000個格納される．そこで，この配列aに0から3までの整数がそれぞれ何個ずつ格納されているか数える処理を，プログラム中の枠内に記入せよ．その際，数えた結果は配列bに格納し，n =0, 1, 2, 3について，配列aに格納されている整数 n の個数がb[n]によって表されるようにせよ．ただし，配列aの内容を変更してはいけない．また，プログラム中ですでに宣言されている変数以外に新たに変数を宣言してはいけない．

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

int main()
{
  int i;
  int a[1000], b[4];

  for (i = 0; i < 1000; i ++) {    // (X)
    a[i] = rand() % 4;
  }
```

```
  for (i = 0; i < 4; i ++) {
    printf("%d %d\n", i, b[i]);
  }
  return 0;
}
```

2．下に示したC言語のプログラムにおいて，行(Y)から書かれているforループでは，配列bに0から9までの整数の乱数が4個格納される．そこで，この配列bの内容にもとづいて，配列aの内容を次のように設定する処理を，プログラム中の枠内に記入せよ．すなわち，配列aの先頭から整数0がb[0]個並び，その直後に整数1がb[1]個並び，またその直後に整数2がb[2]個並び，さらにその直後に整数3がb[3]個並び，そして残りの要素はすべて−1となるように配列aの内容を設定する処理を，プログラム中の枠内に記入せよ．ただし，配列bの内容を変更してはいけない．また，プログラム中ですでに宣言されている変数以外に新たに変数を宣言してはいけない．

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

int main()
{
  int i, j, k;
  int a[40], b[4];

  for (i = 0; i < 4; i ++) {    // (Y)
    b[i] = rand() % 10;
  }
```

```

```

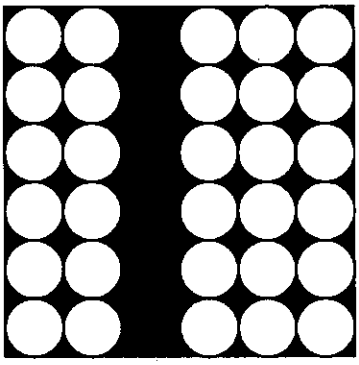
```
  for (i = 0; i < 40; i ++) {
    printf("%d ", a[i]);
  }
  printf("\n");
  return 0;
}
```

3．あなたは，計算機に接続された電光掲示板を操作するC言語のプログラムを書くよう依頼された．この電光掲示板には，横方向に6列，縦方向に6行，合計36個のLEDが格子状に並んでいる．

最初の状態では，36個すべてのLEDが消灯している．必要な環境設定は済んでおり，あなたがプログラム中でLED(x, y)という関数を呼び出しさえすれば，左からx列目，上からy行目のLEDが点灯する．その際，xとyは1,2,3,4,5,6のうちいずれかの値をとらなければならない．

なお，一度点灯させたLEDを消灯させることはできない．また，同じxとyの値について関数LED(x, y)を複数回呼び出してもよいが，その場合は単に左からx列目，上からy行目のLEDがそのまま点灯し続けるだけである．

例えば，右の図Aのように左から3列目を除くすべてのLEDを点灯させるには，以下のようなプログラムを書けばよい．

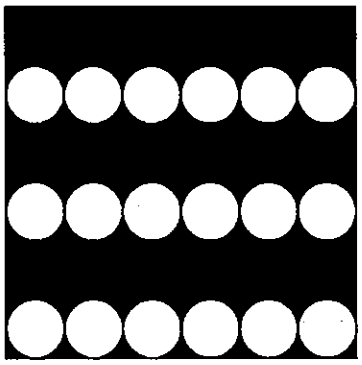


図A

```
for (i = 1; i <= 6; i ++) {
  if (i != 3) {
    for (j = 1; j <= 6; j ++) {
      LED(i, j);
    }
  }
}
```

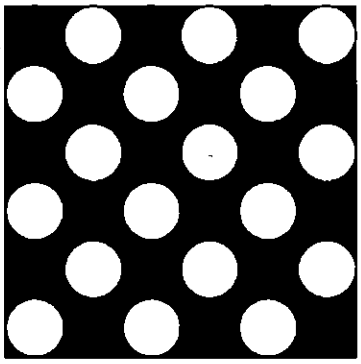
上の例のように，あなたはLEDを点灯させる部分のプログラムを書くことができるだけである．関数を自分で作成することもできないし，#define のようなプリプロセッサ指令も使えない．また，あなたが使用できる変数は，int型のiとjという2つの変数だけであり，新たに変数を宣言することもできない．さらに，LEDを点灯させる関数は，上の例のようにfor文やwhile文などの繰り返し処理の内部で呼び出すものとし，プログラムの文面に「LED」という文字列は2回までしか書いてはいけないとする．このとき，以下の（1）から（4）までの問いに答えよ．

（1）右の図BのようにLEDを点灯させるには，どのようなプログラムを書けばよいか．下の枠内に解答を記入せよ．



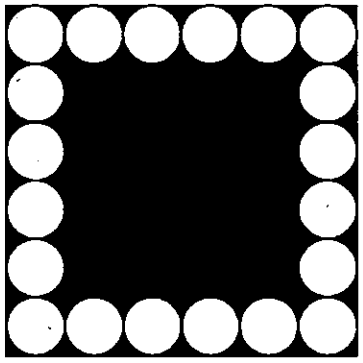
図B

（2）右の図CのようにＬＥＤを点灯させるには，どのようなプログラムを書けばよいか．下の枠内に解答を記入せよ．



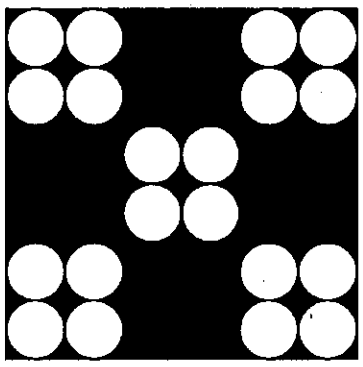
図C

（3）右の図DのようにＬＥＤを点灯させるには，どのようなプログラムを書けばよいか．下の枠内に解答を記入せよ．



図D

（4）右の図ＥのようにＬＥＤを点灯させるには，どのようなプログラムを書けばよいか．下の枠内に解答を記入せよ．



図E

4. あるプロセス（これ以降，このプロセスをプロセス A と呼ぶ）がインターネットからのデータを受信しようとし，その直後に別のアプリケーションプログラムから生成されたプロセス（これ以降，このプロセスをプロセス B と呼ぶ）もインターネットからのデータを受信しようとしたとする．すでに 1 秒が経過したがまだどちらもデータを受信できていない．

この設定の基で，以下の（1）～（10）の設問に答えよ．

(1) 現在 2 つのプロセスはどちらも同じ状態にある．その状態の名称を枠内に解答せよ．

(2) プロセス B の現状をもっともよく説明するものを以下の選択肢 A～D から 1 つ選び，枠内に解答せよ．

A: CPU の状態がカーネルモードになるのを待っている．
B: プロセス A に与えられたタイムスライスが終わるのを待っている．
C: CPU スケジューラまで処理が進んでおり，状況が変化するまでそこで止められている．
D: （システムコール）処理モジュールまで処理が進んでおり，状況が変化するまでそこで止められている．

(3) プロセス A およびプロセス B 以外にはプロセスが存在しないとする．このコンピュータの状況をもっともよく説明するものを以下の選択肢 A～D から 1 つ選び，枠内に解答せよ．

A: この 1 秒間，CPU は休みなく OS を実行していた．
B: この 1 秒間，CPU はほとんどなにも実行していない．
C: この 1 秒間，CPU は 2 つのプロセスを同時に実行し続けていた．
D: この 1 秒間，CPU は周期的かつ間欠的に 2 つのプロセスを実行していた．

(4) インターネットからパケットが受信装置 (Network Interface Card=NIC と呼ばれることが多い）に届いた．この時，割込みは起きるか，枠内に解答せよ．

(5) 受信されるパケットは IPv4 であれば例えば以下のように構成される．空欄 A～E に対するもっとも適切なものを選択肢から選び，それぞれの枠内に解答せよ（選択肢は 1 回のみ使用できる）．

1. 以下の情報から構成される [A] ヘッダ
   - バージョン，それぞれ長さ [B] の送信元および宛先アドレス，プロトコルなど
2. 以下の情報から構成される [C] ヘッダ
   - 送信元および宛先 [D] 番号，[E] 番号，ウィンドウサイズなど
3. データ本体

【選択肢】4 バイト，8 バイト，8 ビット，16 バイト，HTTP，IP，PCB，TCP，シーケンス，ポート，ページ

A:	B:	C:	D:	E:

(6)　一般的なクライアント PC での受信処理において，受信された（宛先がこのコンピュータのアドレスであった）パケットのデータ本体がどのプロセスに送られるかを決めるのは，主にパケット中のどの情報か．枠内に解答せよ．

(7)　プロセス A は OS 経由で受信したデータをそのメモリ内に保持し加工する．データとメモリの関係について，もっとも適切な説明を以下の選択肢 A〜D から 1 つ選び，枠内に解答せよ．

A：データ長は有限なので，必ずコード領域内に置かれる（=コード領域が使われる）．
B：ローカル変数として確保した領域に保持されるならスタック領域内に置かれる．
C：32〜64 ビット CPU ならばページテーブル，それ以外なら仮想アドレス内に置かれる．
D：グローバル変数として静的に確保した領域に保持されるならヒープ領域内に置かれる．

(8)　プロセス A のみが受信を繰り返し全てのデータをメモリ内に保持し続けたところ，ある時点で主記憶をちょうど使い切った．第 2 世代以降の OS（Windows や Linux などが含まれる）だとすると，パケットを待ち続けていたプロセス B が使っていたメモリはこの時点でどうなっていると考えられるか，枠内に解答せよ．

(9)　以下の 6 項目 a ～ f を OS がある状況に対応する際に実行される順に並び替え，以下の枠内に解答せよ．

a：CPU スケジューラ，b：（システムコール）処理モジュール，c：ディスパッチャ，
d：ユーザモードへの切換え，e：割込み，f：割込みハンドラ

→　→　→　→　→

(10)　一般に，以下の状況 A〜C において割込みは発生するかしないか，それぞれの枠内に解答せよ．

A：新規ファイルのハードディスク装置への書き込み保存が終わった場合（時点）．
B：C 言語のプログラム中の while 文の繰り返し条件判定式が成立しなくなった場合（時点）．
C：機械語のオペランドで指定された番地が主記憶上にあるが CPU のキャッシュ上にない場合．

A:	B:	C: