**1.** 2 つの符号なし 2 進整数 $X$ と $Y$ を加算し，その和 $Z$ を符号なし 2 進整数として出力する回路について考える．$X$，$Y$，$Z$ はいずれも 4 ビットとする．

(1) $Z$ が表現できる最大の値を 10 進数で示せ．

解答欄

(2) オーバーフローを生じることなく $Z$ に (1) で求めた値が出力されるような $X$，$Y$ への入力値の与え方は全部で何通りあるか答えよ．

解答欄

(3) オーバーフローを生じることなく $Z$ を出力できるような $X$，$Y$ への入力値の与え方は全部で何通りあるか答えよ．

解答欄

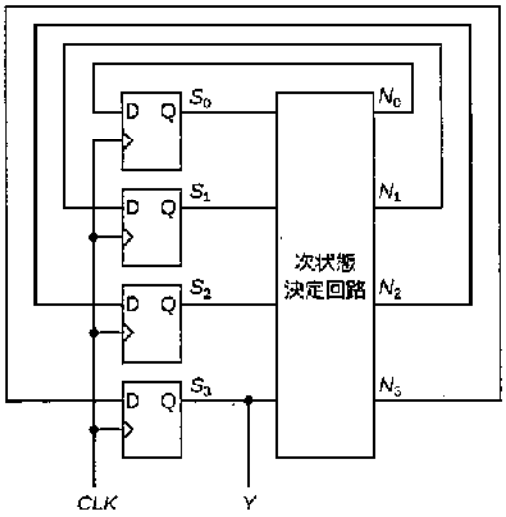
(4) 回路の仕様を変更し，負数を 2 の補数で表す 4 ビットの 2 進整数で $X$，$Y$，$Z$ を表すこととした．このとき，オーバーフローを生じることなく $Z$ を出力できるような $X$，$Y$ への入力値の与え方は全部で何通りあるか答えよ．

解答欄

2. 次の図に示すように，クロック入力 $CLK$ と 1 ビットの出力 $Y$ をもつ順序回路について考える．次状態決定回路には，論理式

$$N_0 = \overline{S_3} \cdot \overline{S_0}$$
$$N_1 = \overline{S_1} \cdot S_0 + S_3 \cdot \overline{S_1} + \overline{S_3} \cdot S_1 \cdot \overline{S_0}$$
$$N_2 = \overline{S_2} \cdot S_1 \cdot S_0 + S_2 \cdot \overline{S_1} + \overline{S_3} \cdot S_2 \cdot \overline{S_0} + S_3 \cdot \overline{S_1}$$
$$N_3 = S_3 \cdot \overline{S_2} + S_2 \cdot S_1 \cdot S_0$$

が組み合わせ回路として実現されている．4 つの D フリップフロップの出力は，いずれも初期状態では 0 であるとする．




(1) $N_0$ の論理式を加法標準形で表せ．ここで，加法標準形とは論理式を最小項の和として表現したものをいう．

解答欄

(2) $CLK$ にクロックパルス信号を入力すると，$Y$ には周期的な信号が出力される．その周期のクロックサイクル数を求めよ．

解答欄

3. メモリアクセスの時間的局所性と空間的局所性の違いについて説明せよ.

解答欄

4. TCP/IP ネットワークのインターネット層における動的ルーティングと静的ルーティングの違いについて説明せよ.

解答欄

5．　以下の問いに答えよ．

（1）実行中のプロセスはオペレーティングシステム内において，①実行状態，②実行可能状態，③待ち状態，のいずれかになる．この時①～③の説明として最も適した選択肢を以下の a~c の中から選べ．また，①から③への状態変化，②から①への状態変化のそれぞれを起こす契機として最も適切な選択肢を d~g の中から選べ．

a. CPU リソースさえ確保できれば，プロセスを実行可能な状態
b. CPU リソースを含む実行に必要なすべてのリソースが確保され，プロセスを実行している状態
c. CPU 以外に実行に必要なリソースが確保されていない状態
d. 自プロセスに割り当てられた CPU 利用時間を使い切ったとき
e. CPU スケジューラによって CPU リソースが割り当てられたとき
f. CPU 以外の実行に必要なリソースを確保できたとき
g. プロセスがシステムコールを実行したとき

**解答欄**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| ① | | ② | | ③ | | ①→③ | | ②→① | |

（2）以下の文章の(A)~(F)の空欄に適切な語を，選択肢から選び解答欄に記入せよ．

複数のプロセスが同時に実行されている場合，共通のリソース（例えば共有変数）にアクセスする場合は(A)が必要となる．また，(A)が必要となる領域のことを(B)と呼ぶ．(A)の解決方法の代表的な例は(C)である．これは，待ち行列と整数変数を持つデータ構造であり，(D)命令と(E)命令により制御する．一般的には(B)に入る前に(D)命令を，出る前に(E)命令を実行する．さらに，より抽象度の高い(A)の方法としてオブジェクト指向の枠組みを用いた(F)がある．

**選択肢**：クリティカルセクション，Dekker のアルゴリズム，デッドロック，モニタ，排他制御，プロセス間通信，テストアンドセット命令，ロック／キー機構，主記憶管理，セマフォ，P, V, スタベーション

**解答欄**

| | | | | | |
|---|---|---|---|---|---|
| A | | B | | C | |
| D | | E | | F | |

（3）あるプロセスにより参照されるページ番号が以下であったとする．

0→1→2→3→1→4→5→3→6→7→6→3→8

主記憶装置のページ枠が 4 であるとき，LRU アルゴリズムを用いた場合のページフォールト回数を答えよ．ただし初期状態では主記憶装置にどのページも読み込まれていないとする．

**解答欄**

（4）ファイルシステムの FAT16 では，管理用の領域を除くディスク全体を$2^{16} = 65536$個のクラスタと呼ばれる単位に分割して管理する．FAT16 で管理用の領域を除いた容量が 2GB のディスクを管理する場合，1 クラスタのサイズを求めよ．また，すべてのファイルのサイズが 8KB でこのシステムに最大限のファイル数を格納した場合のディスクの実質的な使用効率を求めよ．ここで 1KB=1024B, 1MB=1024KB, 1GB=1024MB とする．

**解答欄**

6．決定性有限オートマトン $M$ の初期状態は $q_0$，受理状態は $q_5$ であり，動作は表 1 の状態遷移表で表される．以下の設問に答えよ．

表 1

| 状態＼入力 | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| $q_0$ | $q_0$ | $q_5$ | $q_5$ | $q_2$ |
| $q_1$ | $q_4$ | $q_1$ | $q_5$ | $q_5$ |
| $q_2$ | $q_2$ | $q_5$ | $q_4$ | $q_5$ |
| $q_3$ | $q_0$ | $q_3$ | $q_5$ | $q_5$ |
| $q_4$ | $q_4$ | $q_5$ | $q_5$ | $q_2$ |
| $q_5$ | $q_5$ | $q_5$ | $q_5$ | $q_5$ |

(1) 次の記号列 $w_1$〜$w_5$ のうち，$M$ が受理するものをすべて選び，○で囲め．
解答欄

$w_1 = adac$　　$w_2 = abcd$　　$w_3 = cabc$　　$w_4 = dcaa$　　$w_5 = aadd$

(2) 初期状態 $q_0$ からどのように遷移しても到達できない状態をすべて示せ．
解答欄

(3) 初期状態 $q_0$ と等価な状態をすべて示せ．
解答欄

(4) 受理する言語が $M$ と同じで，状態の数が最小の決定性有限オートマトンを求め，その状態遷移図を示せ．
解答欄

(5) $M$ が受理する言語 $L(M)$ の補集合を受理する決定性有限オートマトンの状態遷移図を示せ．
解答欄

7．以下の記述が正しければ○，誤っていれば×を左の □ に記入せよ．

□ 正規言語はすべてプッシュダウンオートマトンで受理できる．

□ チューリングマシンで受理できる言語はすべて文脈自由である．

□ 図 1 のオートマトン $M$ が受理する言語 $L(M)$ の要素は 1 つである．

□ 決定可能でない正規言語が存在する．

□ $n$ 個の状態をもつ有限オートマトンが，長さ $n$ の記号列を受理するならば，
そのオートマトンが受理する言語は無限個の記号列からなる．

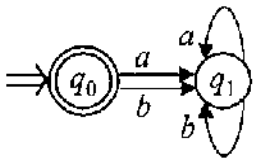


図 1

8．非終端記号の集合を $N = \{S, X\}$，終端記号の集合を $\Sigma = \{a, b\}$，
書き換え規則の集合を $P = \{S \to XX,\ X \to XXX,\ X \to a,\ X \to bX,\ X \to Xb\}$，開始記号を $S$ とする．
文法 $G = \langle N, \Sigma, P, S \rangle$ について，以下の設問に答えよ．

(1) 次の記号列 $w_1$～$w_5$ のうち，文法 $G$ で導出できるものをすべて選び，○で囲め．
**解答欄**

$w_1 = bbabb$　　$w_2 = aabaa$　　$w_3 = ababb$　　$w_4 = babbb$　　$w_5 = aaaaa$

(2) 4 回以下の書き換えで導出できる記号列をすべて記せ．
**解答欄**

(3) 文法 $G$ は曖昧であることを示せ．
**解答欄**

9. ユークリッド互除法を用いて，最大公約数を求めるプログラムを C 言語で作成する．ユークリッド互除法の説明を読み，空欄を埋めてプログラムを完成させよ．ただし，新たな変数は追加しないものとする．

**[ユークリッド互除法とは]**

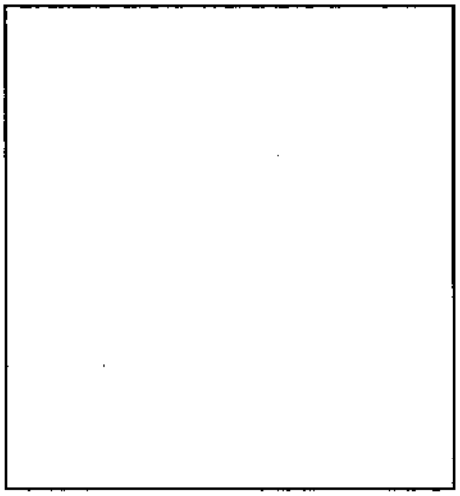
2 つの自然数 a, b (a ≧ b) について，a の b による剰余を r とすると，a と b との最大公約数は b と r との最大公約数に等しいという性質が成り立つ．この性質を利用して，b を r で割った剰余，除数 r をその剰余で割った剰余，と剰余を求める計算を逐次繰り返すと，剰余が 0 になった時の除数が a と b との最大公約数となる．

<例> 1071 と 1029 の最大公約数を求める
1071 を 1029 で割った剰余は 42
1029 を 42 で割った剰余は 21
42 を 21 で割った剰余は 0
よって，最大公約数は 21

**[プログラム(C 言語)]** (注：プログラム中の¥n は改行コード(\n と同じ))

```
#include <stdio.h>
int main(void)
{
    int a, b, r;
    r = 1;
    // 最大公約数を求める 2 つの自然数を変数 a, b に入力する．
    // ただし，a >= b とする．
    printf("Input two integers ==> ");
    scanf("%d %d", &a, &b);
    // ユークリッド互除法で最大公約数を求める
    // ここで，r は剰余
    while (r > 0) {
        [          ]
    }
    // 最大公約数を出力する
    printf("The greatest common divisor is %d.¥n", a);
    return 0;
}
```

**解答欄**



10. 次のプログラムの説明を読んで，(1)~(4)の空欄を埋めて C 言語のプログラムを完成させよ．ただし，新たな変数は追加しないものとする．

**[プログラムの説明]**

1~3 の数値を入力すると，デジタル時計のようにその数値をデジタル文字で表示するプログラムを作成する．入力は正の整数とし，0 が入力された場合プログラムを終了し，範囲外の数値が入力された場合は，その数値を無視し再度入力を繰り返す．

| デジタル文字 1 | デジタル文字 2 | デジタル文字 3 |
|---|---|---|
| □■□ | ■■■ | ■■■ |
| □■□ | □□■ | □□■ |
| □■□ | ■■■ | ■■■ |
| □■□ | ■□□ | □□■ |
| □■□ | ■■■ | ■■■ |

デジタル文字は 5 行 3 列とし，白(□)と黒(■)で構成する．デジタル文字のパターンは，下記に示す 3 次元配列 pat で保持する．そして，配列の各要素には，白(□)で表示する部分には 0，黒(■)で表示する部分には 1 を格納する．

pat[表示する数値-1][文字パターンの行][文字パターンの列]

例えば，デジタル文字 2 を場合，pat の値は下記のようになる．

■■■ pat[1][0][0] = 1, pat[1][0][1] = 1, pat[1][0][2] = 1
□□■ pat[1][1][0] = 0, pat[1][1][1] = 0, pat[1][1][2] = 1
■■■ pat[1][2][0] = 1, pat[1][2][1] = 1, pat[1][2][2] = 1
■□□ pat[1][3][0] = 1, pat[1][3][1] = 0, pat[1][3][2] = 0
■■■ pat[1][4][0] = 1, pat[1][4][1] = 1, pat[1][4][2] = 1

**[プログラム(C 言語)]** (注：プログラム中の¥n は改行コード(\n と同じ))

```
#include <stdio.h>
int main(void)
{
    int pat[3][5][3];
    int i, j, k, num = 1;

// 次ページへ続く
```

```
// 表示パターンを保持する変数を初期化
for (i = 0; i < 3; i++) {
    for ( (1) ) {
        for (k = 0; k < 3; k++) {
            pat[i][j][k] = 0;
        }
    }
}

// デジタル文字１の表示パターン設定
for (j = 0; j < 5; j++) {
    (2)
}

// デジタル文字２の表示パターン設定
for (k = 0; k < 3; k++) {
    pat[1][0][k] = 1;
    pat[1][2][k] = 1;
    pat[1][4][k] = 1;
}
pat[1][1][2] = 1;
pat[1][3][0] = 1;

// デジタル文字３の表示パターン設定
for (k = 0; k < 3; k++) {
    pat[2][0][k] = 1;
    pat[2][2][k] = 1;
    pat[2][4][k] = 1;
}
(3)
```

**解答欄**

| | |
|---|---|
| (1) | |
| (2) | |
| (3) | |

// 次ページへ続く

```
    while (num != 0) {
        printf("数値を入力してください(1～3, 0 で終了): ");
        scanf("%d", &num);
        if (num >= 1 && num <= 3) {
            //  num に対応するデジタル文字を出力する.
            //  なお, 白(□)と黒(■)の出力には, それぞれ下記を利用すること
            //  白(□)の出力 : printf("□");
            //  黒(■)の出力 : printf("■");
```

| (4) |
|---|

```
        }
    }
    printf("Exit Program¥n");
    return 0;
}
```

**解答欄**

| | |
|---|---|
| (4) | |