ＶＢグラフィックス：生産システムの問題（ガントチャートの作成）

担当：諏訪・川野・寒川

ガントチャート(Gantt chart)とは：

ガントが考案した管理図表で,時間を区切った図表に作業の開始時から完了時までバーで示し,作業の先行関係や総処理時間などを視覚的に表したもの。

フローショップスケジューリング(Flow-shop Scheduling)とは：

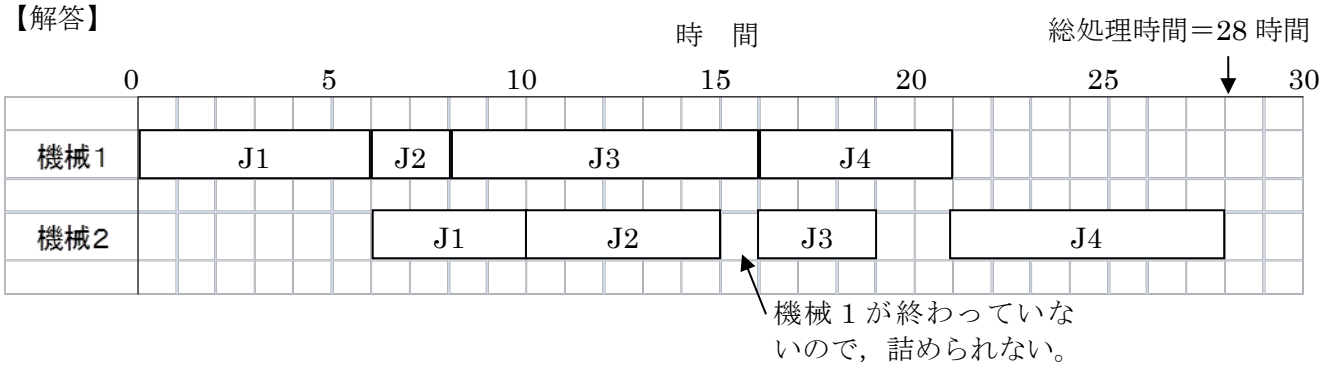
すべてのジョブ（各製品づくり）において，機械の利用順序が同じで，各機械は同時に２つ以上の仕事ができない生産形態。

【例題】表１に示すようなフローショップ問題において，製品の生産順序が J1→J2→J3→J4 の場合のガントチャートを描き，総処理時間を求めよ。

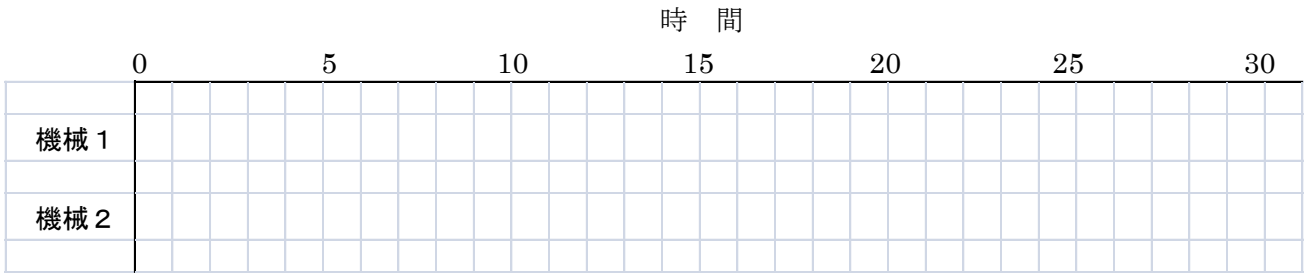
表１　各機械の加工時間 単位 :時間）

	製品（J1）	製品（J2）	製品（J3）	製品（J4）
機械１	6	2	8	5
機械２	4	5	3	7

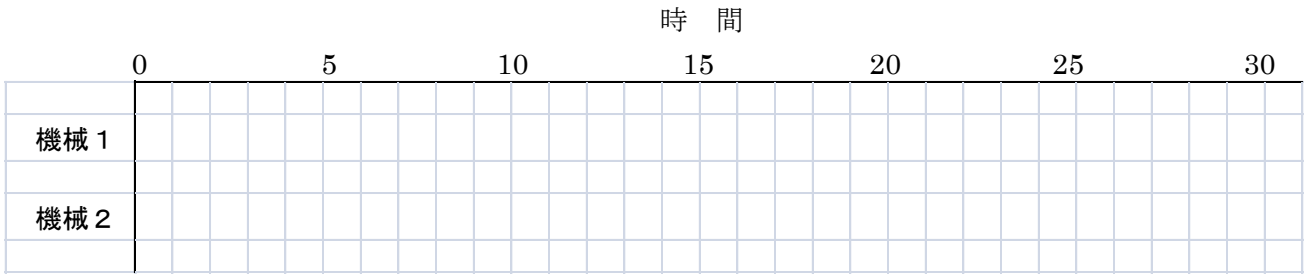
【解答】



【問題１】J2→J4→J1→J3 の場合のガントチャートを描き，総処理時間を求めよ。（手書きでよい）

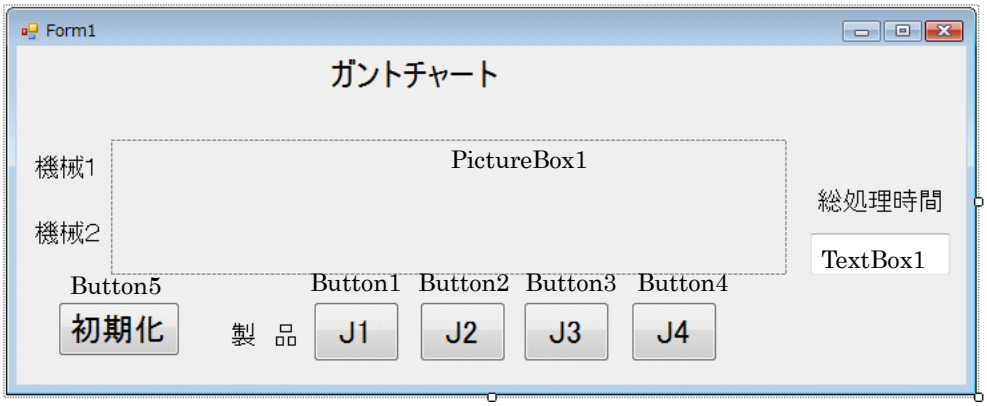


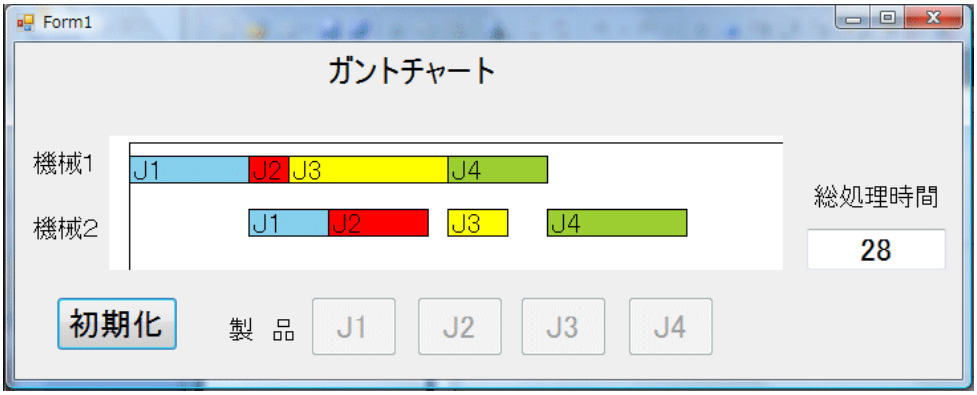
【問題２】J3→J1→J4→J2 の場合のガントチャートを描き，総処理時間を求めよ。



演習 G３ 〔gantt〕

下の図のように，PictureBox，TextBox，Button，Label を用意する。「初期化」ボタンを押して，ガントチャートの基盤を PictureBox に描き，変数の初期化を行う。次に製品ボタンを任意の順番に押すとガントチャートが作成され，総処理時間が TextBox に表示されるプロジェクト。





【プログラムコード】

```
Public Class Form1
    Dim j1( ) As Integer = {0, 6, 4}   ←  j1( )は「配列（番号付き変数）」 j1(0)=0, j1(1)=6, j1(2)=4
    Dim j2( ) As Integer = {0, 2, 5}
    Dim j3( ) As Integer = {0, 8, 3}   } 各機械の加工時間
    Dim j4( ) As Integer = {0, 5, 7}
    Dim m1last As Integer   ←  各製品を配置したときの機械１の終了時刻
    Dim m2last As Integer   ←  各製品を配置したときの機械２の終了時刻
    Dim x, y, w, h As Integer   ←  ガントチャートのバー（四角形）を描くための座標と幅と高さ
    Dim flag, count As Integer   ←  ボタンを最初に押したことを記録するフラグと押した回数
```

初期化

```
    Private Sub Button5_Click(sender As System.Object,  ・・・
        Dim g As Graphics = PictureBox1.CreateGraphics( )   ←  PictureBox1 にグラフィックス作成
        g.Clear(Color.White)              ←  ピクチャーボックスを白色で埋める
        g.DrawLine(Pens.Black, 15, 15 - 10, 15, 200)   ←  ガントチャートの縦軸を描く
        g.DrawLine(Pens.Black, 15, 15 - 10, 850, 15 - 10)   ←  ガントチャートの横軸を描く
        Button1.Enabled = True      ←  ボタン１を押せる状態にする
```

```
        Button2. Enabled = True      ←　ボタン２を押せる状態にする
        Button3. Enabled = True      ←　ボタン３を押せる状態にする
        Button4. Enabled = True      ←　ボタン４を押せる状態にする
        TextBox1.Text = ""   ←　テキストボックスをクリアする
        m1last = 0
        m2last = 0          } 変数の初期化
        flag = 0
        count = 0
    End Sub
```

ガントチャートの作成

```
Private Sub Button①_Click(sender As System.Object,・・・
        Dim g As Graphics = PictureBox1.CreateGraphics( ) ←　PictureBox1 にグラフィックス作成
        Dim objFont = New Font("ＭＳ　Ｐゴシック", 16) ←　グラフィックス用にフォントを定義
機械1のバー描画
        x = 15 + m1last * 15   ←　バーの左上の x 座標（現時点の終了時刻 m1last）
        y = 15
        w = j①(1) * 15   ←　バーの幅（J1 の加工時間 j1(1)　）
        h = 20
        g.FillRectangle(Brushes.SkyBlue, x, y, w, h)
        g.DrawRectangle(Pens.Black, x, y, w, h)
        g.DrawString("J①", objFont, Brushes.Black, x, y)
        m1last = m1last + j①(1)   ←　J1 の配置が終わると，新しい終了時刻に更新

        If flag = 0 Then
            m2last = j①(1)      } 押したボタンが１個目ならば，機械２の終了時刻の初期値は
            flag = 1              機械１の終了時刻となる
        End If

        If m2last < m1last Then
            m2last = m1last     } 機械１が終わってからでないと機械２を利用できないので，
        End If                      機械１の終了時刻の方が大きいなら，それが m2last になる。
機械2のバー描画
        x = 15 + m2last * 15
        y = 15 + 40
        w = j①(2) * 15
        h = 20
        g.FillRectangle(Brushes.SkyBlue, x, y, w, h)
        g.DrawRectangle(Pens.Black, x, y, w, h)
        g.DrawString("J①", objFont, Brushes.Black, x, y)
        m2last = m2last + j①(2)    ←　J1 の配置が終わると，新しい終了時刻に更新

        Button①.Enabled = False   ←　ボタン１が押せない状態にする
        count = count + 1      ←　押したボタンの個数をカウント
        If count = 4 Then
            TextBox1.Text = m2last     } ボタンをすべて押したら，総処理時間を表示
        End If
    End Sub
```

以下，ボタン 2～ボタン 4 をクリックしたときのプロシジャーは，上記コード中の①をボタン番号に変える。

バーの色は，Brushes.Red（赤），Brushes.Yellow（黄色），Brushes.YellowGreen（黄緑）などを用いる。

【補足】最適アルゴリズム

　　総処理時間が最小になるアルゴリズム（手順）は，いろいろ考案されているが，代表的なものに「ジョンソン法」がある。

ジョンソン法

１．最小加工時間を選択する
２．この時間が前工程（機械１）であれはそのジョブを前から並べる
３．この時間が後工程（機械２）であればそのジョブを後ろから並べる。
４．未割り当てジョブが無くなるまでに上記３ステップを繰り返す。

このアルゴリズムを VB でコーディングすれば，自動的にスケジュールが決定される。