ภาคผนวก ก

ภาคผนวก ก

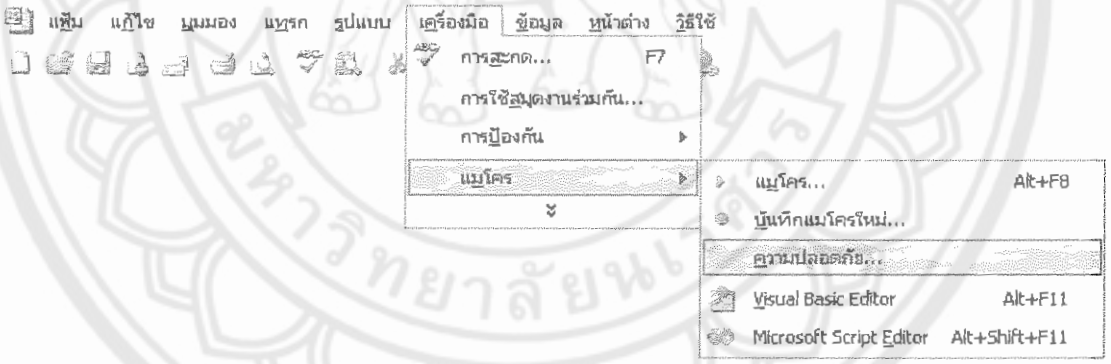
คู่มือแนะนำการใช้โปรแกรม

โปรแกรมที่เราสร้างขึ้นในการแก้ปัญหาวิธีการจัดลำดับการทำงานของเครนด้วยวิธีการอบอ่อนจำลอง มีชื่อว่า SA Crane Scheduling จะเป็นโปรแกรมที่ช่วยหารูปแบบในการจัดเรียงลำดับการทำงานของเครนท่าเรือ เพื่อรูปแบบการจัดเรียงที่ทำให้เวลาที่ใช้ในกระบวนการทำงานทั้งหมดมีค่าน้อยที่สุด ซึ่งในส่วนนี้จะอธิบายขั้นตอนการใช้งานโปรแกรม และการป้อนค่าต่างๆ ที่โปรแกรมใช้ในการประมวลผล

1. การเปิดใช้งานโปรแกรม

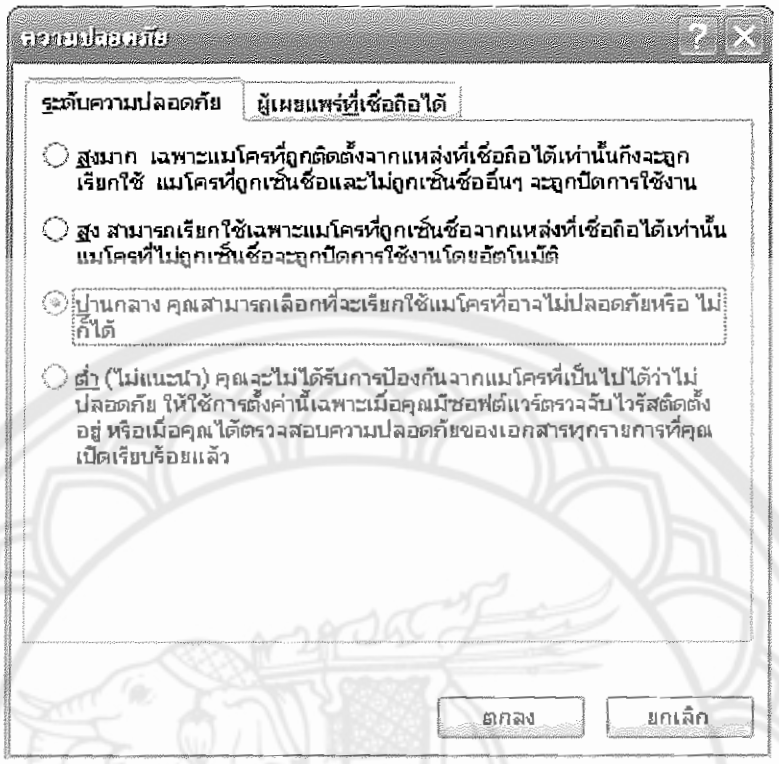
ก่อนที่จะเปิดโปรแกรม SA Crane Scheduling นี้มาใช้งานได้นั้น ในครั้งแรกของเครื่องคอมพิวเตอร์แต่ละเครื่อง จะต้องมีการตั้งค่าแมโคร ใน Microsoft Excel ก่อน ซึ่งสามารถทำได้ดังนี้

1.1 เริ่มจากการเปิด Microsoft Excel ขึ้นมา ทำการตั้งค่าความปลอดภัย โดยเลือกที่แทบเครื่องมือ เลือกแมโคร และเลือกความปลอดภัย ดังรูป



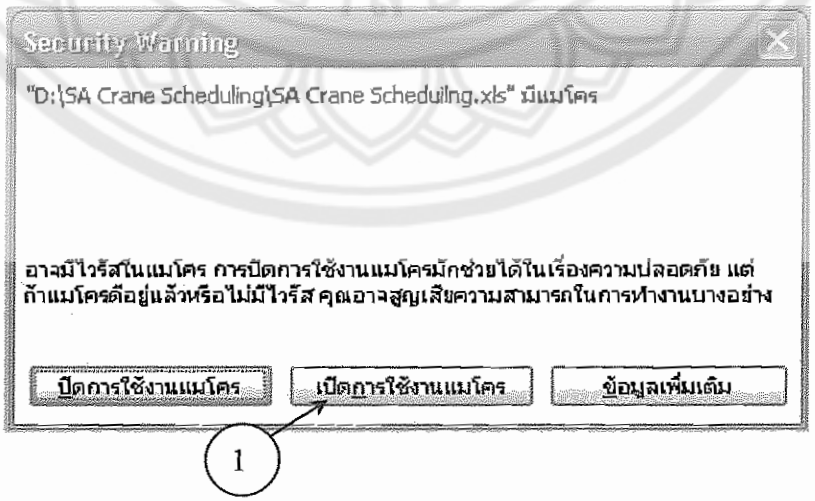
รูปที่ ก-1 แสดงขั้นตอนการตั้งค่าความปลอดภัยแมโคร

จากนั้นทำการเลือกทำการเลือกระดับความปลอดภัยที่ระดับปานกลาง แล้วคลิกปุ่ม ตกลง ดังรูป ก-2 แล้วทำการปิด File Microsoft Excel นี้ก่อน



รูปที่ ก-2 แสดงกรอบตอบโต้ความปลอดภัยเพื่อตั้งระดับความปลอดภัยที่ระดับปานกลาง

1.2 เปิด File โปรแกรมชื่อ SA Crane Scheduling นี้ขึ้นมา จะปรากฏหน้าต่าง Security Warning เช่นนี้ขึ้นมา ให้เราเลือก ที่ปุ่ม เปิดการใช้งานแมโคร ที่ตำแหน่ง 1 ดังรูป



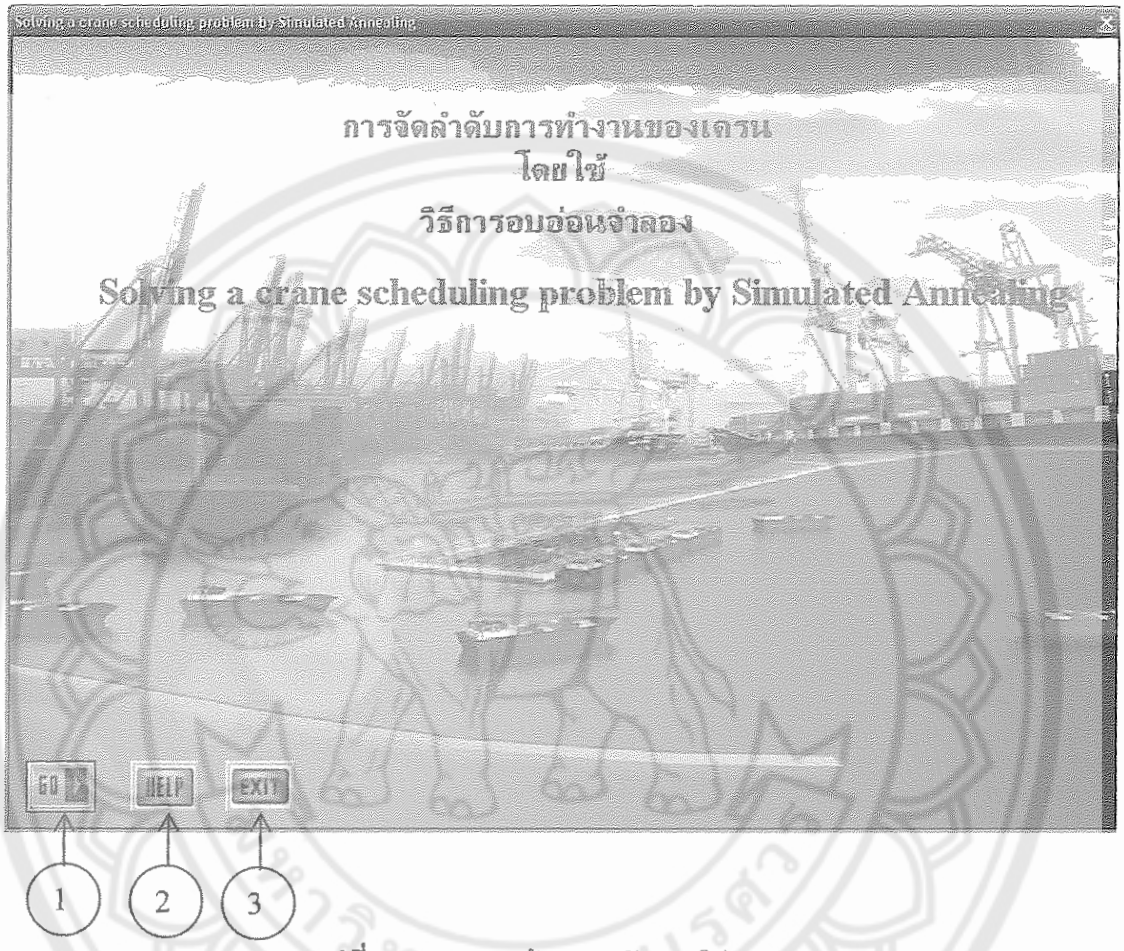
รูปที่ ก-3 แสดงหน้าต่างเพื่อเปิดการใช้งานแมโคร

2. เข้าสู่โปรแกรม SA Crane Scheduling

เมื่อทำการเปิดใช้งานงานแมโครแล้ว จะพบกับหน้าที่ 1 ของโปรแกรมเป็นหน้าที่แสดงข้อความต้อนรับเข้าสู่โปรแกรม และโปรแกรมจะปิดหน้านี้อัตโนมัติ

รูปที่ ก- 4 หน้าที่ 1 ของโปรแกรมแสดงข้อความต้อนรับเข้าสู่โปรแกรม

เมื่อหน้าต้อนรับของโปรแกรมปิดแล้ว จะเข้าสู่หน้าเมนูหลัก ที่เป็นหน้าที่ 2 ของโปรแกรม แสดง รูปที่ ก- 5



รูปที่ ก- 5 แสดงหน้าเมนูหลักของโปรแกรม

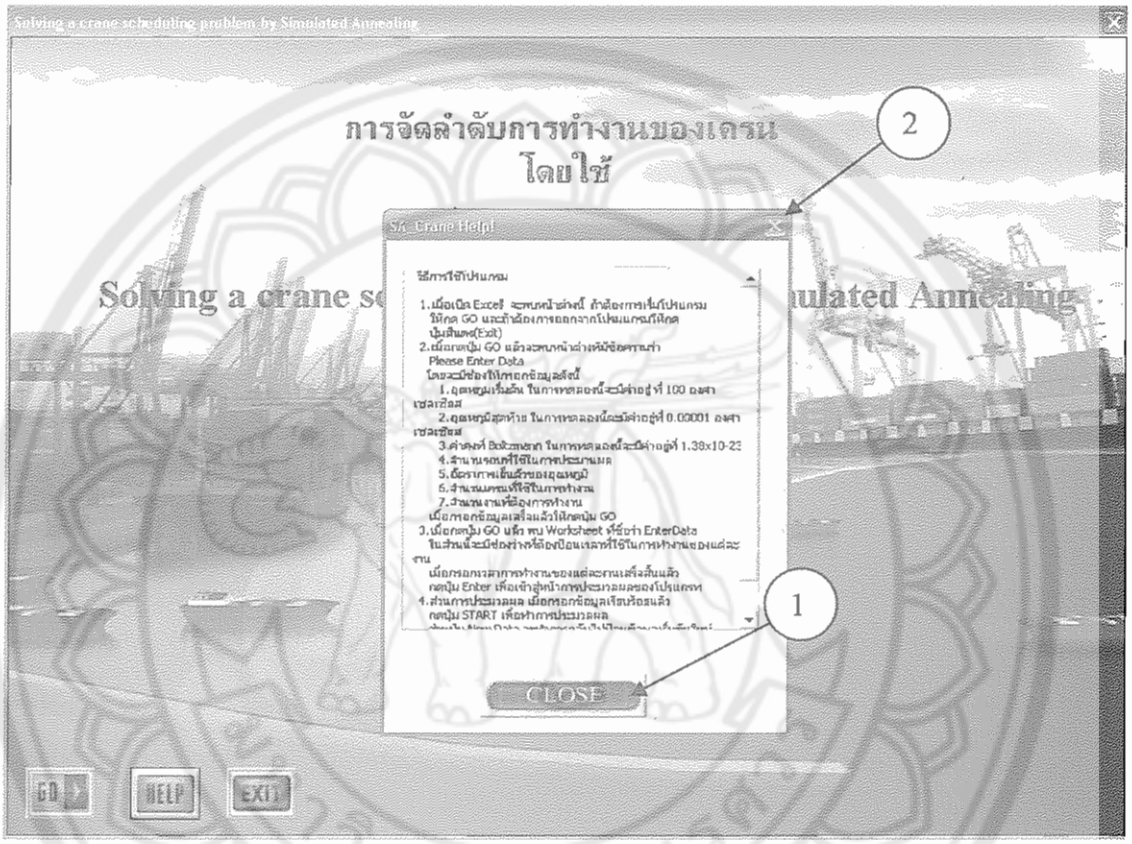
ในหน้าเมนูหลักนี้จะมี ปุ่ม และแต่ละปุ่มมีหน้าที่ดังต่อไปนี้

หมายเลข 1 ปุ่ม เมื่อผู้ใช้คลิกปุ่มนี้ จะนำเข้าสู่ส่วนรับข้อมูลในหน้าที่ 3 ของโปรแกรม ดังรูปที่ ก-7

หมายเลข 2 ปุ่ม เมื่อผู้ใช้คลิกปุ่มนี้ จะแสดงกล่องข้อความที่อธิบายค่าที่ใช้ป้อนในการประมวลผล และขั้นตอนการทำงานของโปรแกรม

หมายเลข 3 ปุ่ม เมื่อผู้ใช้คลิกปุ่มนี้ จะทำการปิดการใช้งานโปรแกรม

จากรูปที่ ก- 5 ถ้ากด ปุ่ม จะแสดงกล่องข้อความ SA_Crane Help! เพื่อแนะนำการใช้โปรแกรมอธิบายค่าที่ใช้ป้อนในการประมวลผล และขั้นตอนการทำงานของโปรแกรม โดยสังเขป แสดงดังรูปที่ ก-6 ดังนี้

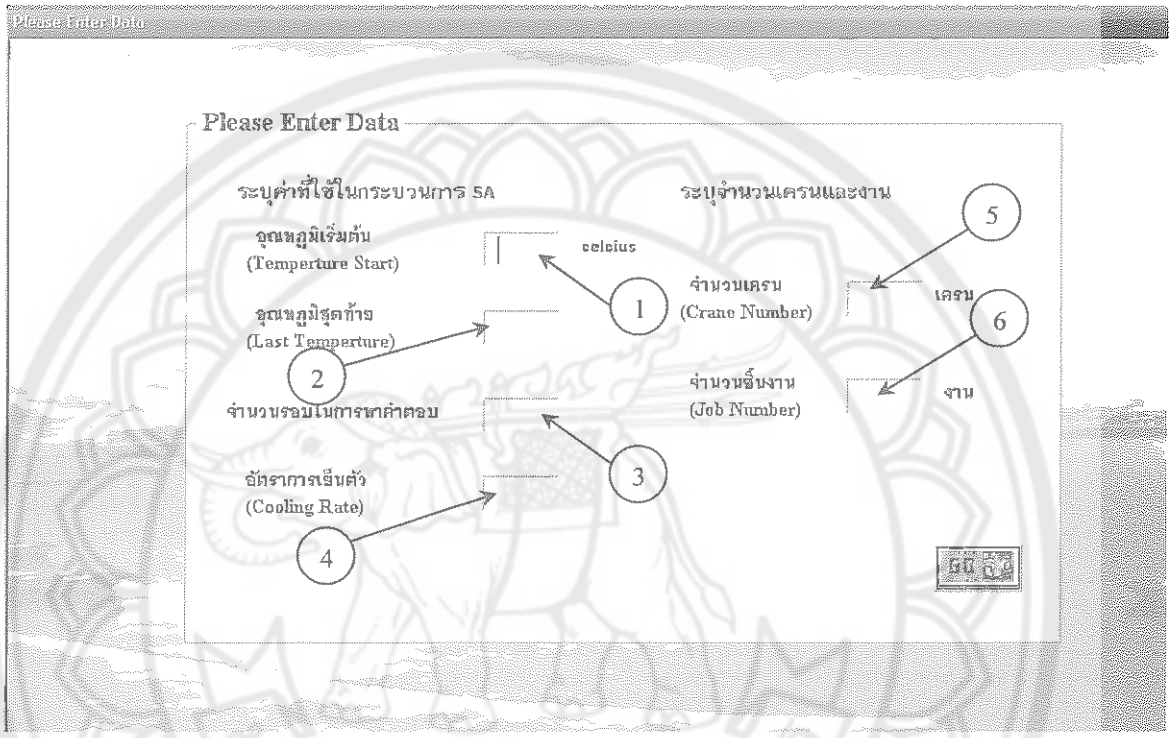


รูปที่ ก- 6 หน้าต่างกล่องข้อความช่วยเหลือ เมื่อกดปุ่ม HELP

เมื่อต้องการปิดหน้าต่างนี้สามารถทำได้ 2 วิธี คือ 1) ให้กดปุ่ม CLOSE ในตำแหน่งหมายเลข 1 หรือ 2) กดปุ่มปิด ที่ ตำแหน่งหมายเลข 2 ได้เช่นกัน

3. ส่วนรับข้อมูลของโปรแกรม

จากรูปที่ ก- 5 ถ้ากดปุ่ม จะแสดงฟอร์ม Please Enter Data เพื่อให้กรอกข้อมูล ดังรูปที่ ก- 7



รูปที่ ก- 7 หน้าที่ 3 ของโปรแกรมเป็นส่วนรับข้อมูลของโปรแกรมที่ผู้ใช้ต้องกรอก

โดยจะแสดงขึ้นมาเพื่อให้กรอกข้อมูล มีข้อมูล 2 ส่วนที่ผู้ใช้ต้องกรอก คือ

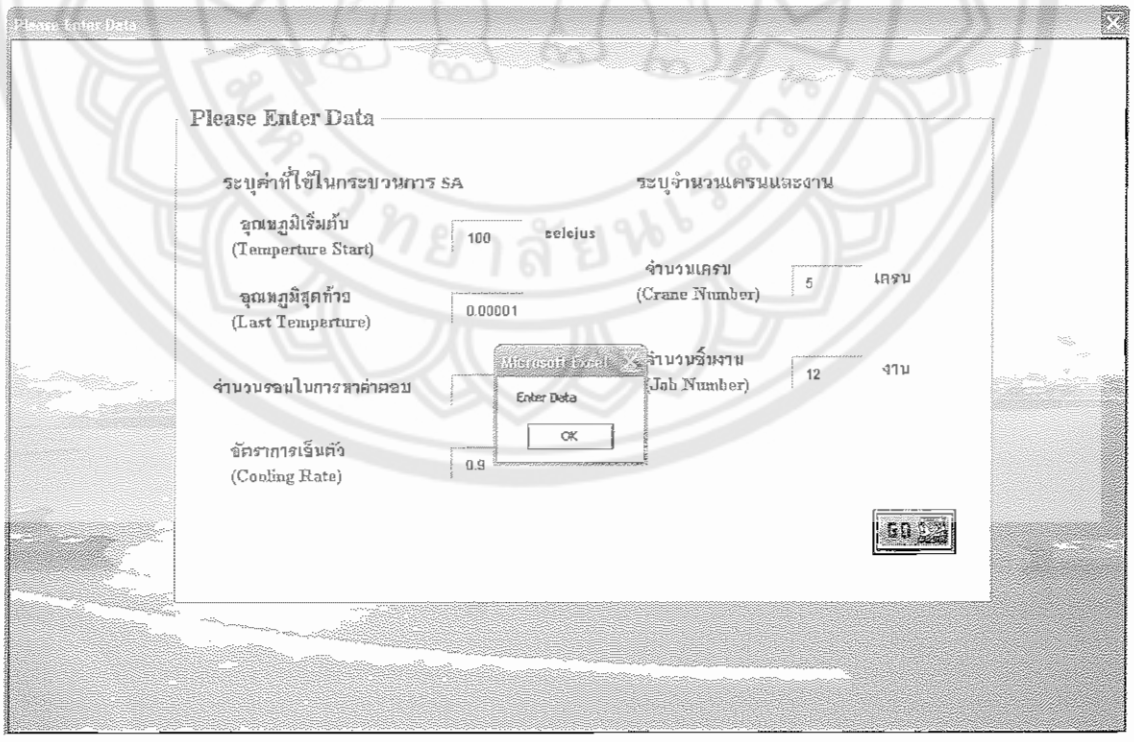
ส่วนที่ 1 ข้อมูลในกระบวนการอบอ่อนจำลอง ตั้งแต่ช่องหมายเลข 1-4

ส่วนที่ 2 ข้อมูลที่เป็นจำนวนเครน และจำนวนงาน ที่ต้องการหาการจัดเรียง ในช่องที่ 5 และ6

ข้อมูลที่ผู้ใช้ต้องกรอกเข้าไปในส่วนรับข้อมูล

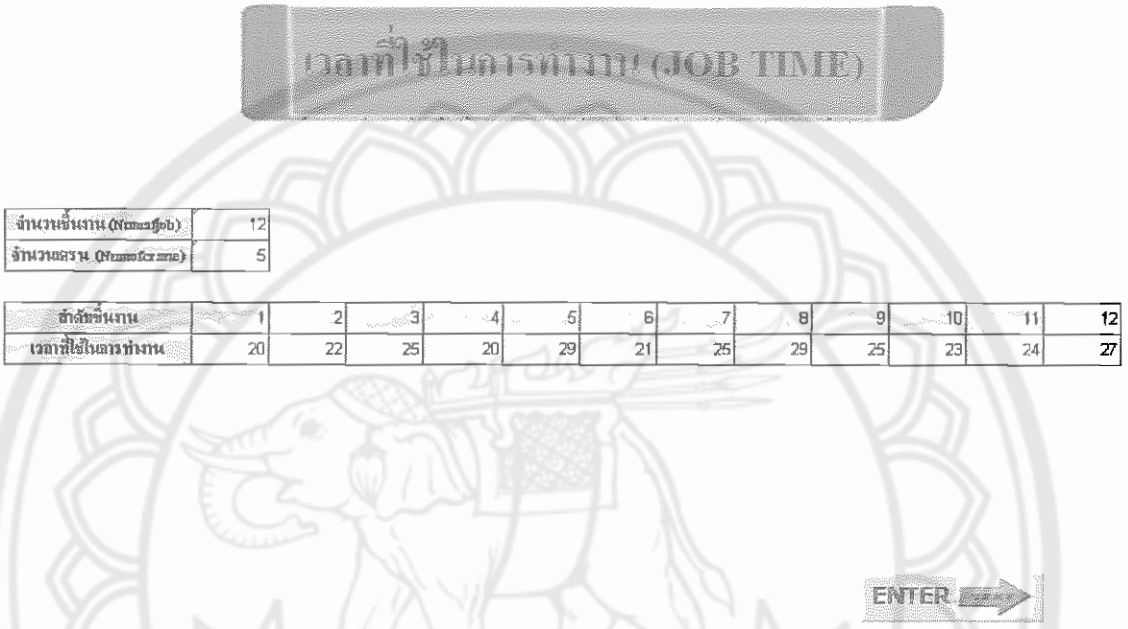
1. **อุณหภูมิเริ่มต้น ที่ตำแหน่งหมายเลข 1 ผู้ใช้จะทดลองกรอกอุณหภูมิเริ่มต้นเท่าไรก็ได้ตามแต่ผู้ใช้ต้องการจะทดสอบผลการทดลองโปรแกรม**
2. **อุณหภูมิสุดท้าย ที่ตำแหน่งหมายเลข 2 ผู้ใช้สามารถกรอกค่าอุณหภูมิ ได้ความความ** ต้องการในการทดลอง แต่ก็ควรจะต้องมีค่าที่ น้อยกว่าอุณหภูมิเริ่มต้นเสมอ
3. จำนวนรอบในการหาคำตอบ ในตำแหน่งหมายเลข 3 เป็นค่าที่ผู้ใช้ต้องการให้โปรแกรมประมวลผลตามรอบที่กำหนด
4. อัตราการเย็นตัว ในตำแหน่งหมายเลข 4 เป็นค่าที่ไม่เกิน 0.99 หรือมากกว่า แต่ต้องไม่ถึง 1
5. จำนวนเครน ในตำแหน่งหมายเลข 5 เป็นข้อมูลของจำนวนเครนที่ใช้ในการทำงาน
6. จำนวนชิ้นงาน ในตำแหน่งหมายเลข 6 เป็นจำนวนงานที่ต้องการทำ

และผู้ใช้ต้องทำการกรอกข้อมูลให้ครบทุกช่อง ถ้าไม่เช่นนั้นโปรแกรมจะไม่สามารถทำงาน**ต่อ**ไปได้ และจะมีกล่องข้อความเตือน Enter Data ดังรูปที่ ก-8 ผู้ใช้ต้องกรอกข้อมูลให้ครบโปรแกรมจึงจะทำงานต่อไปได้



รูปที่ ก- 8 แสดงกล่องข้อความเตือนเมื่อผู้ใช้กรอกข้อมูลไม่ครบ

จากฟอร์ม Please Enter Data เมื่อผู้ใช้กรอกข้อมูลในส่วนรับข้อมูลครบแล้ว ให้กดปุ่ม เพื่อเข้าสู่หน้าต่อไปของโปรแกรม เป็นหน้าที่ 4 ของโปรแกรม ซึ่งจะเป็นหน้าที่ผู้ใช้ต้องกรอกค่าเวลาที่ ใช้ในการทำงานแต่ละงาน



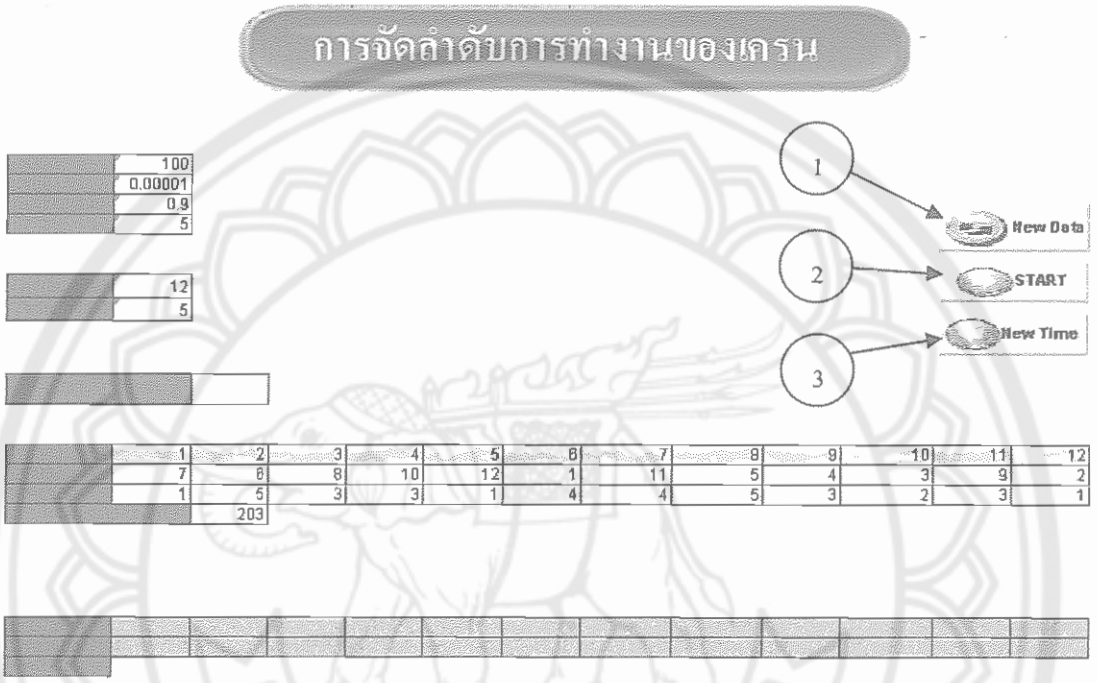
รูปที่ ก- 9 หน้าที่ 4 ของโปรแกรมแสดงส่วนรับข้อมูลเวลาในการทำงานแต่ละงาน

ค่าเวลาที่ผู้ใช้ต้องกรอกใน หน้าที่ 4 ของโปรแกรม เป็นค่าเวลาที่ใช้ในการทำงานแต่ละงานซึ่งโปรแกรมจะไม่จำกัดเวลาที่ใช้ในการทำงาน ขึ้นอยู่กับผู้ใช้ว่าต้องการทำงานที่มีเวลาในการทำงานงานเป็นเท่าไร

หลังจากที่ผู้ใช้ได้กรอกค่าเวลาที่ใช้ในการทำงานแต่ละงาน ในหน้าที่ 4 ของโปรแกรมเสร็จแล้ว ให้กด ปุ่ม เพื่อเข้าสู่หน้าที่ 5 ของโปรแกรม จะทำการประมวลผลต่อไป ดัง **รูปที่** ก-10

4.ส่วนแสดงผลการทดลอง

เมื่อทำการกรองข้อมูลในหน้าที่ 4 ของโปรแกรม เสร็จสิ้นแล้วจะเข้าสู่หน้าที่ 5 ของโปรแกรม ซึ่งเป็นหน้าที่ทำการประมวลผล และแสดงผลการทำงานของโปรแกรม



รูปที่ ก- 10 หน้าที่ 5 ของโปรแกรมแสดงตำแหน่งของปุ่มสำหรับใช้งาน

หน้าที่ 5 ของโปรแกรมนี้ มี 3 ปุ่ม ซึ่งมีหน้าที่ดังนี้

หมายเลข 1 ปุ่ม

ใช้ปุ่มนี้ เมื่อต้องการกรอกข้อมูลเริ่มต้นใหม่ แล้วฟอร์ม Please Enter Data ดังรูปที่ ก-7 จะแสดง เพื่อให้ผู้ใช้กรอกข้อมูลใหม่อีกครั้ง

หมายเลข 2 ปุ่ม

ใช้ปุ่มนี้เมื่อ จะแสดงรูปแบบในการจัดเรียงแต่ละแบบพร้อมทั้งเวลาที่ใช้ในกระบวนการทำงานทั้งหมดของแต่ละรูปแบบ จะแสดงผลดังรูปที่ ก-11

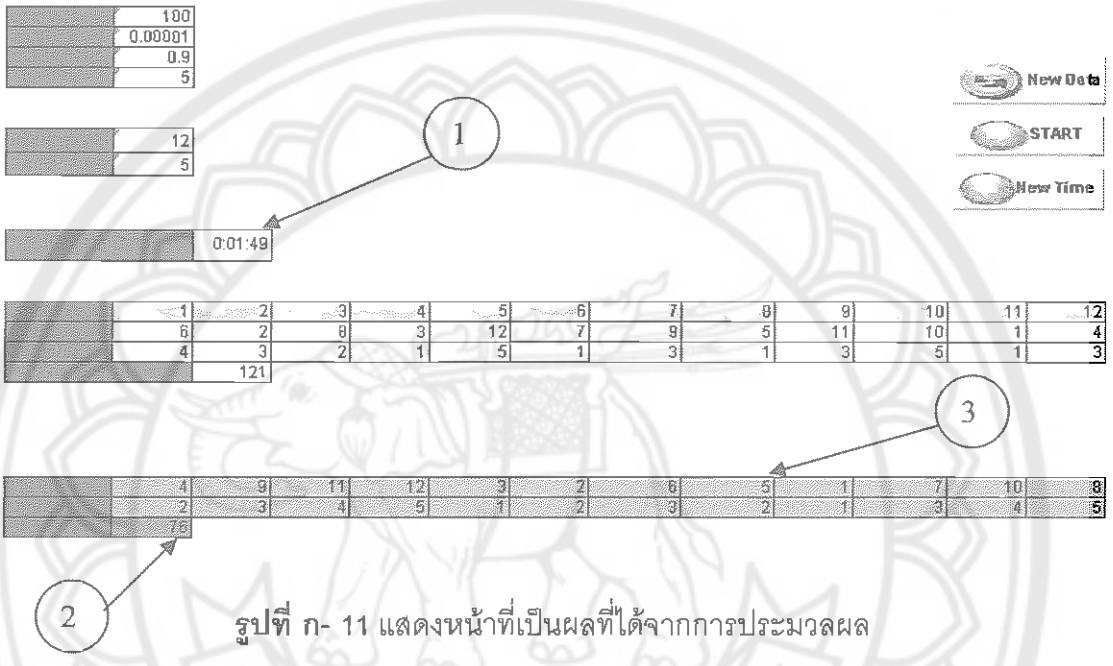
หมายเลข 3 ปุ่ม

ตำแหน่งหมายเลข 3 เป็นปุ่มที่สั่งให้โปรแกรมกลับไปที่หน้าที่ 4 ของโปรแกรม เพื่อให้ผู้ใช้ทำการกรอกค่าเวลาที่ใช้ในการทำงานใหม่ ซึ่งเพราะบางปัญหามีจำนวนเครนที่ใช้ในการทำงาน และจำนวนงานที่ต้องการทำงานเท่ากันแต่ต่างที่ เวลาที่ใช้ในการทำงาน แต่ค่าต่างๆ ที่ใช้ในการประมวลผลยังคงเดิม จึงไม่จำเป็นต้องกรอกค่าใหม่ทั้งหมดทำให้สะดวกต่อการใช้โปรแกรมมากยิ่งขึ้น

เมื่อกดปุ่ม

และโปรแกรมประมวลผลเสร็จเรียบร้อยจะแสดงผลดังนี้

การจัดลำดับการทำงานของเครน



รูปที่ ก- 11 แสดงหน้าที่เป็นผลที่ได้จากการประมวลผล

หมายเลข 1 คือ เวลาที่โปรแกรมประมวลผลเสร็จสมบูรณ์ (Run Time)

หมายเลข 2 คือ เวลาที่ที่เครนทำงานเสร็จสิ้นของคำตอบที่ดีที่สุด

หมายเลข 3 คือ ลำดับการจัดลำดับ งานที่ทำ และเครนที่ถูกทำ ตามลำดับ

ภาคผนวก ข

ภาคผนวก ข

ในส่วนนี้จะกล่าวถึงโค้ดที่ใช้ในการเขียนโปรแกรม

1. ชื่อโพซีเยอร์ของโปรแกรมทั้งหมดและหน้าที่ ของโปรแกรมทั้งหมด

ตารางที่ ข-1 แสดงโค้ดที่สั่งให้ หน้าแรกของโปรแกรมแสดง

```
Private Sub UserForm_Activate()
   Application.wait Now + TimeValue("0:00:02")
   Unload Me
End Sub
```

ตารางที่ ข-2 แสดงสั่งการทำงานของ หน้า UserForm1

```
Private Sub CommandButton1_Click()
" คำสั่งให้เช็คการกรอกข้อมูลของผู้ใช้
If Crane = "" Or WorkNum = "" Or SATemp = "" Or Round = "" Or SABolt = "" Or SARate = "" Or
SALast = "" Then
      MsgBox " Enter Data"
ElseIf SARate > 0.99999999999 Then
   MsgBox "Cool Rate could be not Over 0.9"
ElseIf SATemp < SALast Then
   MsgBox "The initial temperature must more than the last temperature"
Else
      Call FristNSClear                    " คั่งสั่งให้ล้างข้อมูลเก่าในตำแหน่งที่ระบุออก

" คำสั่งกำหนดค่าต่างๆ ให้แสดงในตำแหน่งที่กำหนด
      Worksheets("Solve").Cells(9, 3).Value = SATemp
      Worksheets("Solve").Cells(10, 3).Value = SALast
      Worksheets("Solve").Cells(11, 3).Value = SARate
```

```
Worksheets("Solve").Cells(12, 3).Value = SABolt
Worksheets("Solve").Cells(13, 3).Value = Round
Worksheets("EnterData").Cells(8, 5).Value = WorkNum
Worksheets("EnterData").Cells(9, 5).Value = Crane
Worksheets("Solve").Cells(15, 3).Value = WorkNum
Worksheets("Solve").Cells(16, 3).Value = Crane

NumofCrane = Crane
NumofJob = WorkNum

 For i = 1 To NumofJob
    Worksheets("EnterData").Cells(11, 4 + i).Value = ""
    Worksheets("EnterData").Cells(12, 4 + i).Value = ""
    Worksheets("EnterData").Cells(11, 4 + i).Borders.Weight = xlThin
    Worksheets("EnterData").Cells(12, 4 + i).Borders.Weight = xlThin
    Worksheets("EnterData").Cells(11, 4 + i).Value = i
    Worksheets("EnterData").Cells(11, 4 + i).Interior.Color = RGB(235, 200, 238)
    Worksheets("EnterData").Cells(12, 4 + i).Interior.Color = RGB(186, 250, 211)
 Next i

 For i = 1 To NumofJob
    Worksheets("Solve").Cells(22, 2 + i).Value = i
    Worksheets("Solve").Cells(22, 2 + i).Interior.Color = RGB(235, 200, 238)
    Worksheets("Solve").Cells(22, 2 + i).Borders.Weight = xlThin
    Worksheets("Solve").Cells(24, 2 + i).Interior.Color = RGB(186, 250, 211)
    Worksheets("Solve").Cells(24, 2 + i).Borders.Weight = xlThin
    Worksheets("Solve").Cells(23, 2 + i).Interior.Color = RGB(186, 250, 211)
Worksheets("Solve").Cells(23, 2 + i).Borders.Weight = xlThin
 Next i
```

```
    Worksheets("EnterData").Activate        " คำสั่งให้แสดง  Worksheets("EnterData")
    Unload Me                               "คำสั่งให้ซ่อนตัวเอง
End If
End Sub
```

ตารางที่ ข-3 แสดงโค้ดที่สั่งให้ส่วนที่กำหนด รับเฉพาะข้อมูลตัวเลขเท่านั้น

```
Private Sub Crane_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
Call checkNumeric(KeyAscii)
End Sub

Private Sub Round_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
Call checkNumeric(KeyAscii)
End Sub

Private Sub SABolt_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
Call checkNumeric(KeyAscii)
End Sub
Private Sub SALast_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
Call checkNumeric(KeyAscii)
End Sub

Private Sub SARate_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
    Call checkNumeric(KeyAscii)
End Sub
Private Sub SATemp_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
   Call checkNumeric(KeyAscii)
End Sub
Private Sub WorkNum_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
```

```
    Call checkNumeric(KeyAscii)
End Sub
```

ตารางที่ ข-4 แสดงโค้ดสั่งการทำงานของ หน้า UserForm2

```
Private Sub CommandButton1_Click()              "เพื่อให้แสดงกล่องข้อความ Help
UserForm3.Show
End Sub

Private Sub CommandButtonExit_Click()           "คำสั่งเพื่อออกจากโปรแกรม
ThisWorkbook.Close
End Sub

Private Sub CommandButtonEnter_Click()          " เข้าสู่ UserForm1 เพื่อกรอกข้อมูล
UserForm1.Show
End Sub
```

ตารางที่ ข-5 แสดงโค้ดสั่งการทำงานของ หน้า UserForm3 เพื่อแนะนำการใช้โปรแกรมในส่วน Help

โค้ดแสดงสั่งการทำงานของ หน้า UserForm3 เพื่อแนะนำการใช้โปรแกรมในส่วน Help

```
Private Sub UserForm_Initialize()
Dim htlpText As String
totalRow = Worksheets("WellCome").Cells(65536, 1).End(xlUp).Row
   For i = 350 To totalRow
       helpText = helpText & Worksheets("WellCome").Cells(i, 1).Text & vbCrLf
                                        "....helpText = ข้อความที่อยู่ใน Range A
   Next i
```

```
With Label1
  .Caption = helpText
  .AutoSize = True
End With
   Frame1.ScrollHeight = Label1.Height
   Frame1.ScrollTop = 0
End Sub
```

ตารางที่ ข-6 แสดงโค้ดประกาศตัวแปรใน Module

```
Dim i As Integer
Dim JobNum As Integer
Dim Boltmann As Double
Dim CraneNum As Integer
Dim CoolRate As Integer
Dim Temp As Integer
Dim Prop As Double
Dim R As Double
Dim R1 As Integer
```

ตารางที่ ข-7 แสดงโค้ดคำสั่งที่ใช้ล้างข้อมูลเก่าก่อนทำการประมวลผลใหม่ใน Sheets" Solve"

```
Function FristNSClear()
   Worksheets("Solve").Range("C22:AZ24").ClearFormats
   Worksheets("Solve").Range("C22:AZ24").ClearContents
   Worksheets("Solve").Cells(25, 4).ClearContents
   Worksheets("EnterData").Range("E11:Z12").ClearFormats
   Worksheets("EnterData").Range("E11:Z12").ClearContents
   Worksheets("Solve").Range("C30:AZ31").ClearFormats
```

```
    Worksheets("Solve").Range("C30:AZ31").ClearContents
    Worksheets("Solve").Cells(32, 3).ClearContents
    Worksheets("Solve").Cells(32, 3).ClearFormats
    Worksheets("Solve").Range("A40:AZ500").ClearFormats
Worksheets("Solve").Range("A40:AZ500").ClearContents
Worksheets("Solve").Cells(39, 1).ClearContents
Worksheets("Solve").Cells(19, 4).ClearContents
End Function
```

ตารางที่ ข-8 แสดงโค้ดคำสั่งที่ใช้ในการกำหนดให้เป็นค่าตัวเลขจำนวนเต็มเท่านั้น

```
Function checkNumeric(KeyAscii)
    If KeyAscii < 46 Or KeyAscii = 47 Or KeyAscii > 57 Then KeyAscii = 0
End Function
```

ตารางที่ ข-9 คำสั่งที่ทำการสุ่มการจัดเรียงใหม่

```
Sub Newns()
NumofJob = Worksheets("EnterData").Cells(8, 5)
NumofCrane = Worksheets("EnterData").Cells(9, 5)

 Dim iArr() As Integer
      ReDim iArr(1 To NumofJob)
      For i = 1 To NumofJob
           iArr(i) = i
      Next i
      For i = NumofJob To 1 Step -1
           R = Int(Rnd() * (i - 1)) + 1
           Temp = iArr(R)
```

```
            iArr(R) = iArr(i)
            iArr(i) = Temp
        Next i

        For i = 1 To NumofJob
            Worksheets("Solve").Cells(23, 2 + i) = iArr(i)
        Next i

        For i = 1 To NumofJob
            Worksheets("Solve").Cells(24, 2 + i) = 0
        Next i
            R1 = Int((NumofJob - 1) * Rnd() + 1)
            Worksheets("Solve").Cells(24, 2 + R1) = 1

For i = 2 To NumofCrane
            Do
                R1 = Int((NumofJob - 1) * Rnd() + 1)
            Loop Until Worksheets("Solve").Cells(24, 2 + R1) = 0
            Worksheets("Solve").Cells(24, 2 + R1) = i
          Next i
          For i = 1 To NumofJob
            If Worksheets("Solve").Cells(24, 2 + i) = 0 Then
                  Worksheets("Solve").Cells(24, 2 + i) = Int((NumofCrane * Rnd()) + 1)
            End If
          Next i
End Sub
```

ตารางที่ ข-10 แสดงโค้ดกำหนดไม่ให้แสดง grid

```
Private Sub Worksheet_Activate()
ActiveWindow.DisplayGridlines = False
End Sub
```

ตารางที่ ข-11แสดงโค้ดประกาศตัวแปรใน Sheet2 (EnterData)

```
Dim TotalNS As Integer
Dim NumofJob As Integer
Dim NumofCrane As Integer
Dim Ptime() As Integer
Dim BestRow As Integer
Dim CurSolJob() As Integer
Dim CurSolCrane() As Integer
```

ตารางที่ ข-12 แสดงโค้ดคำสั่งให้เก็บค่าเวลาที่ใช้ในการทำงานแต่ละงานไว้

```
Public Sub ReadPTime()
Dim i As Integer
Dim j As Integer
For i = 1 To NumofJob
    Ptime(i) = Cells(4, 2 + i)
Next i
End Sub
```

ตารางที่ ข-13 แสดงโค้ดคำสั่งที่ทำการสุ่มหารูปแบบการจัดเรียงแบบต่างจากต้นแบบ

```
Sub FunctionNS()
Dim Newns As Integer
Dim z As Integer
NumofJob = Cells(15, 3)
NumofCrane = Cells(16, 3)
TotalNS = 2 * (NumofJob - 1)
Dim Tnewns() As Integer
Dim CurSolJob() As Integer
Dim CurSolCrane() As Integer
ReDim CurSolJob(1 To NumofJob)
ReDim CurSolCrane(1 To NumofCrane)
ReDim Tnewns(1 To 2 * (NumofJob - 1) - 2)
Dim NSSolJob() As Integer
Dim NSSolCrane() As Integer
ReDim NSSolJob(1 To TotalNS, 1 To NumofJob)
ReDim NSSolCrane(1 To TotalNS, 1 To NumofJob)
Dim Temp As Integer
Dim Temp1 As Integer
Dim Temp2 As Integer
Dim R1 As Integer
Dim R2 As Integer
Dim i As Integer, j As Integer, k As Integer, m As Integer, n As Integer, R As Integer, F As
Integer
Dim iArr() As Integer
ReDim iArr(1 To NumofJob)
For i = 1 To TotalNS
   For j = 1 To NumofJob
      NSSolJob(i, j) = Cells(23, 2 + j).Value
```

```
    Next j
Next i
For i = 1 To TotalNS
    For j = 1 To NumofJob
        NSSolCrane(i, j) = Cells(24, 2 + j).Value
    Next j
Next i

F = Int(Rnd() * (NumofJob - 2) + 2)
Temp1 = NSSolJob(1, 1)

For i = 1 To F - 1
    NSSolJob(1, i) = NSSolJob(1, i + 1)
Next i

NSSolJob(1, F) = Temp1

For j = 1 To NumofJob
    Cells(40, 2 + j) = NSSolJob(1, j)
Next j
F = Int(Rnd() * (NumofJob - 2) + 2)
Temp1 = NSSolCrane(1, 1)
For i = 1 To F – 1
    NSSolCrane(1, i) = NSSolCrane(1, i + 1)
Next i

NSSolCrane(1, F) = Temp1
For j = 1 To NumofJob
    Cells(40, 2 + NumofJob + 2 + j) = NSSolCrane(1, j)
```

```
Next j
m = Int((NumofJob - 1) - 1) * Rnd + 1
Temp2 = NSSolJob(TotalNS, m)
For i = m To NumofJob - 1
   NSSolJob(TotalNS, i) = NSSolJob(TotalNS, i + 1)
Next i
NSSolJob(TotalNS, NumofJob) = Temp2
For j = 1 To NumofJob
   Cells(41, 2 + j) = NSSolJob(TotalNS, j)
Next j

m = Int((NumofJob - 1) - 1) * Rnd + 1
Temp2 = NSSolCrane(TotalNS, m)

For i = m To NumofJob - 1
   NSSolCrane(TotalNS, i) = NSSolCrane(TotalNS, i + 1)
Next i
NSSolCrane(TotalNS, NumofJob) = Temp2
For j = 1 To NumofJob
   Cells(41, 2 + NumofJob + 2 + j) = NSSolCrane(TotalNS, j)
Next j
For i = 2 To NumofJob - 1
   R1 = Int((i - 1) - 1) * Rnd + 1
   Temp1 = NSSolJob(2 * i - 2, R1)
   For j = R1 + 1 To i
      NSSolJob(2 * i - 2, j - 1) = NSSolJob(2 * i - 2, j)
   Next j
   NSSolJob(2 * i - 2, i) = Temp1
Next i
```

```
For i = 2 To NumofJob - 1
   R2 = Int(NumofJob - (i + 1)) * Rnd + (i + 1)
   Temp2 = NSSolJob(2 * i - 1, i)
   For j = i + 1 To R2
      NSSolJob(2 * i - 1, j - 1) = NSSolJob(2 * i - 1, j)
   Next j
   NSSolJob(2 * i - 1, R2) = Temp2
Next i

For i = 2 To NumofJob - 1
   R1 = Int((i - 1) - 1) * Rnd + 1
   Temp1 = NSSolCrane(2 * i - 2, R1)
   For j = R1 + 1 To i
      NSSolCrane(2 * i - 2, j - 1) = NSSolCrane(2 * i - 2, j)
   Next j
   NSSolCrane(2 * i - 2, i) = Temp1
Next i

For i = 2 To NumofJob - 1
   R2 = Int(NumofJob - (i + 1)) * Rnd + (i + 1)
   Temp2 = NSSolCrane(2 * i - 1, i)
   For j = i + 1 To R2
      NSSolCrane(2 * i - 1, j - 1) = NSSolCrane(2 * i - 1, j)
   Next j
   NSSolCrane(2 * i - 1, R2) = Temp2
Next i

For i = 2 To TotalNS - 1
```

```
    For j = 1 To NumofJob
        Cells(40 + i, 2 + j) = NSSolJob(i, j)
    Next j
Next i


For i = 2 To TotalNS - 1
    For j = 1 To NumofJob
        Cells(40 + i, 2 + NumofJob + 2 + j) = NSSolCrane(i, j)
    Next j
Next i


For z = 1 To TotalNS
    Call MakespanInterval(z)
Next z


For i = 1 To (TotalNS - 2)


 Tnewns(i) = Cells(39 + i, 2 + NumofJob + 2 + NumofJob + 1).Value
Next i
Newns = Tnewns(1)
For i = 1 To 2 * (NumofJob - 1) - 2
    If Tnewns(i) <= Newns Then
        Newns = Tnewns(i)
        BestRow = i
    End If
Next i
Cells(39 + TotalNS + 1, 3) = Newns
End Sub
```

ตารางที่ ข-14 แสดงโค้ดคำสั่งที่สั่งให้มีการวนรอบในการบวนการทำงานของ SA เพื่อหาคำตอบที่ดีที่สุด

```
Public Sub SACrane()
Dim Tend As Double
Dim Tstart As Double
Dim R As Integer
Dim Rcount As Integer
Dim NSnow As Integer
Dim BestNS As Integer
Dim Tnewns As Double
Dim Newns As Integer
Dim Delta As Double
Dim Prob As Double
Dim U As Double
Dim d As Single
Dim k As Double
Dim BestNew As Integer
Dim T As Double
Dim StartTime
Dim EndTime
Dim RunningTime
StartTime = Time
NumofJob = Cells(15, 3)
NumofCrane = Cells(16, 3)
Call ReadPTime
Tend = Cells(10, 3)
Tstart = Cells(9, 3)
R = Cells(13, 3)
d = Cells(11, 3)
```

```
k = Cells(12, 3)
U = Rnd
NSnow = Cells(25, 4)
T = Tstart
BestNS = NSnow
Do
   R = Cells(13, 3)
   Call Newns1
   Do
      Randomize
      Call FunctionNS
      Rcount = R
      NSnow = Cells(25, 4)
      Newns = Cells(39 + TotalNS + 1, 3)
      If Newns <= NSnow Then
        NSnow = Newns
        For i = 1 To NumofJob
            Cells(31, 2 + i).Interior.Color = RGB(150, 150, 250)
            Cells(31, 2 + i).Borders.Weight = xlThin
            Cells(30, 2 + i).Interior.Color = RGB(150, 150, 250)
            Cells(30, 2 + i).Borders.Weight = xlThin
        Next i
        For i = 1 To NumofJob
          Cells(23, 2 + i) = Cells(39 + BestRow, 2 + i)
          Cells(24, 2 + i) = Cells(39 + BestRow, 2 + NumofJob + 2 + i)
        Next i
        If Newns <= BestNS Then
          BestNS = Newns
          For i = 1 To NumofJob
```

```
            Cells(30, 2 + i) = Cells(39 + BestRow, 2 + i)
            Cells(31, 2 + i) = Cells(39 + BestRow, 2 + NumofJob + 2 + i)
          Next i
        End If
      ElseIf Newns > NSnow Then
        Delta = Newns - NSnow
        Prob = Exp(-Delta / (k * T))
        U = Rnd
        If U <= Prob Then
          NSnow = Newns
          For i = 1 To NumofJob
            Cells(23, 2 + i) = Cells(39 + BestRow, 2 + i)
            Cells(24, 2 + i) = Cells(39 + BestRow, 2 + NumofJob + 2 + i)
          Next i
        End If
      End If
      Cells(25, 4) = NSnow
      Cells(25, 3).Interior.Color = RGB(50, 150, 250)
      Cells(25, 4).Borders.Weight = xlThin
      Cells(32, 3) = BestNS
      Cells(32, 3).Interior.Color = RGB(50, 150, 250)
      Cells(32, 3).Borders.Weight = xlThin
        R = Rcount - 1
    Loop Until R = 0
      T = T * d
Loop Until T <= Tend
For i = 1 To NumofJob
  Cells(31, 2 + i).Interior.Color = RGB(150, 150, 250)
  Cells(31, 2 + i).Borders.Weight = xlThin
```

```
    Cells(30, 2 + i).Interior.Color = RGB(150, 150, 250)
    Cells(30, 2 + i).Borders.Weight = xlThin
Next i
EndTime = Time
RunningTime = EndTime - StartTime

Cells(19, 4) = RunningTime
End Sub
```

ตารางที่ ข-15 แสดงโค้ดคำสั่งที่สั่งให้รวมเวลาในการจัดเรียงแบบต่างๆ ของการจัดเรียงแต่ที่ทำการสุ่มได้ทั้งหมด FunctionNS

```
Public Sub MakespanInterval(z As Integer)
Dim STI() As Integer
Dim FTI() As Integer
Dim STJ() As Integer
Dim FTJ() As Integer
Dim OrderInCrane() As Integer
Dim Now() As Integer
Dim Order() As Integer
Dim Crane() As Integer
ReDim STJ(0 To NumofCrane, 0 To NumofJob)
ReDim FTJ(0 To NumofCrane, 0 To NumofJob)
ReDim STI(0 To NumofCrane, 0 To NumofJob)
ReDim FTI(0 To NumofCrane, 0 To NumofJob)
ReDim OrderInCrane(1 To NumofCrane, 1 To NumofJob)
ReDim Now(1 To NumofCrane)
ReDim Order(1 To NumofJob)
ReDim Crane(1 To NumofJob)
```

```
NumofJob = Cells(15, 3)
NumofCrane = Cells(16, 3)

For i = 1 To NumofJob
    Order(i) = Cells(39 + z, 2 + i)
    Crane(i) = Cells(39 + z, 2 + NumofJob + 2 + i)
Next i

Dim CountCrane()   As Integer
ReDim CountCrane(1 To NumofCrane)

Dim ContInt As Integer
Dim Interval() As Integer
ReDim Interval(1 To NumofJob)

Dim IntTime() As Integer
ReDim IntTime(0 To NumofCrane, 0 To NumofJob)
Const UpBound = 10000
For i = 0 To NumofCrane
    For j = 0 To NumofJob
        STJ(i, j) = 0
        FTJ(i, j) = 0
    Next j
Next i

For i = 1 To NumofCrane
    For j = 1 To NumofJob
        IntTime(i, j) = 0
    Next j
```

```
Next i


For i = 1 To NumofCrane
   CountCrane(i) = 0
Next i


For i = 1 To NumofJob
   CountCrane(Cells(39 + z, 2 + NumofJob + 2 + i)) = CountCrane(Cells(39 + z, 2 +
NumofJob + 2 + i)) + 1
   OrderInCrane(Cells(39 + z, 2 + NumofJob + 2 + i), CountCrane(Cells(39 + z, 2 +
NumofJob + 2 + i))) = Cells(39 + z, 2 + i)
Next i
For i = 1 To NumofJob
   For j = 1 To NumofCrane
      For k = 1 To NumofJob
         STI(j, k) = 0
         FTI(j, k) = 0
      Next k
   Next j
   For j = 1 To NumofCrane
      Interval(j) = 0
      For k = 1 To NumofJob
         IntTime(j, k) = 0
      Next k
   Next j
   Now(Crane(i)) = Now(Crane(i)) + 1
   STJ(Crane(i), Now(Crane(i))) = FTJ(Crane(i), Now(Crane(i)) - 1)
   FTJ(Crane(i), Now(Crane(i))) = STJ(Crane(i), Now(Crane(i))) + Ptime(Order(i))
   For j = 1 To NumofCrane
```

```
If Crane(i) < j Then
   ContInt = 0
   For k = 1 To Now(j)
      If FTJ(j, k) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
         If STJ(j, k) < FTJ(Crane(i), Now(Crane(i)) - 1) Then
            If OrderInCrane(j, k) > Order(i) Then
               Interval(j) = Interval(j) + 1
               STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
               ContInt = 1
               IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
               - FTJ(Crane(i), Now(Crane(i)) - 1) + FTJ(j, k)
               If k = Now(j) Then
                  FTI(j, Interval(j)) = UpBound
                  IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
               End If
            ElseIf OrderInCrane(j, k) < Order(i) Then
               If k = Now(j) Then
                  Interval(j) = Interval(j) + 1
                  STI(j, Interval(j)) = FTJ(j, k)
                  FTI(j, Interval(j)) = UpBound
                  IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
               End If
            End If
         ElseIf STJ(j, k) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
            If FTJ(j, k - 1) < FTJ(Crane(i), Now(Crane(i)) - 1) Then
               If OrderInCrane(j, k) > Order(i) Then
                  Interval(j) = Interval(j) + 1
                  STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
                  ContInt = 1
```

```
            IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
            - FTJ(Crane(i), Now(Crane(i)) - 1) + FTJ(j, k)
            If k = Now(j) Then
              FTI(j, Interval(j)) = UpBound
              IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
            End If
          ElseIf OrderInCrane(j, k) < Order(i) Then
            If STJ(j, k) > FTJ(Crane(i), Now(Crane(i)) - 1) Then
              Interval(j) = Interval(j) + 1
              STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
              IntTime(j, Interval(j)) = STJ(j, k) - FTJ(Crane(i), Now(Crane(i)) - 1)
              FTI(j, Interval(j)) = STI(j, Interval(j)) + IntTime(j, Interval(j))
            End If
            ContInt = 0
            If k = Now(j) Then
              Interval(j) = Interval(j) + 1
              STI(j, Interval(j)) = FTJ(j, k)
              FTI(j, Interval(j)) = UpBound
              IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
            End If
          End If
        ElseIf FTJ(j, k - 1) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
          If OrderInCrane(j, k) > Order(i) Then
            If ContInt = 0 Then
              Interval(j) = Interval(j) + 1
              STI(j, Interval(j)) = FTJ(j, k - 1)
              ContInt = 1
              IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
              - FTJ(j, k - 1) + FTJ(j, k)
```

```
            If k = Now(j) Then
               FTI(j, Interval(j)) = UpBound
               IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
            End If
         ElseIf ContInt = 1 Then
            IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
            - FTJ(j, k - 1) + FTJ(j, k)
            If k = Now(j) Then
               FTI(j, Interval(j)) = UpBound
               IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
            End If
         End If
      ElseIf OrderInCrane(j, k) < Order(i) Then
         If STJ(j, k) > FTJ(j, k - 1) Then
            If ContInt = 0 Then
               Interval(j) = Interval(j) + 1
               STI(j, Interval(j)) = FTJ(j, k - 1)
               FTI(j, Interval(j)) = STJ(j, k)
               IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
            ElseIf ContInt = 1 Then
               IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
               - FTJ(j, k - 1) + STJ(j, k)
               FTI(j, Interval(j)) = STI(j, Interval(j)) + IntTime(j, Interval(j))
               ContInt = 0
            End If
            If k = Now(j) Then
               Interval(j) = Interval(j) + 1
               STI(j, Interval(j)) = FTJ(j, k)
               FTI(j, Interval(j)) = UpBound
```

```
                    IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
                End If
            ElseIf STJ(j, k) = FTJ(j, k - 1) Then
                If ContInt = 1 Then
                    FTI(j, Interval(j)) = STJ(j, k)
                    IntTime(j, Interval(j)) = FTJ(j, k) - STJ(j, k)
                    ContInt = 0
                End If
                If k = Now(j) Then
                    Interval(j) = Interval(j) + 1
                    STI(j, Interval(j)) = FTJ(j, k)
                    FTI(j, Interval(j)) = UpBound
                    IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
                End If
            End If

        End If
    End If
  End If
 End If
Next k
ElseIf Crane(i) > j Then
 ContInt = 0
 For k = 1 To Now(j)
  If FTJ(j, k) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
   If STJ(j, k) < FTJ(Crane(i), Now(Crane(i)) - 1) Then
    If OrderInCrane(j, k) < Order(i) Then
     Interval(j) = Interval(j) + 1
     STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
```

```
ContInt = 1
IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
- FTJ(Crane(i), Now(Crane(i)) - 1) + FTJ(j, k)
If k = Now(j) Then
  FTI(j, Interval(j)) = UpBound
  IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
End If
ElseIf OrderInCrane(j, k) > Order(i) Then
  If k = Now(j) Then
    Interval(j) = Interval(j) + 1
    STI(j, Interval(j)) = FTJ(j, k)
    FTI(j, Interval(j)) = UpBound
    IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
  End If
End If
ElseIf STJ(j, k) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
  If FTJ(j, k - 1) < FTJ(Crane(i), Now(Crane(i)) - 1) Then
    If OrderInCrane(j, k) < Order(i) Then
      Interval(j) = Interval(j) + 1
      STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
      ContInt = 1
      IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
      - FTJ(Crane(i), Now(Crane(i)) - 1) + FTJ(j, k)
      If k = Now(j) Then
        FTI(j, Interval(j)) = UpBound
        IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
      End If
    ElseIf OrderInCrane(j, k) > Order(i) Then
      If STJ(j, k) > FTJ(Crane(i), Now(Crane(i)) - 1) Then
```

```
            Interval(j) = Interval(j) + 1
            STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
            IntTime(j, Interval(j)) = STJ(j, k) - FTJ(Crane(i), Now(Crane(i)) - 1)
            FTI(j, Interval(j)) = STI(j, Interval(j)) + IntTime(j, Interval(j))
          End If
          ContInt = 0
          If k = Now(j) Then
            Interval(j) = Interval(j) + 1
            STI(j, Interval(j)) = FTJ(j, k)
            FTI(j, Interval(j)) = UpBound
            IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
          End If
        End If
      ElseIf FTJ(j, k - 1) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
        If OrderInCrane(j, k) < Order(i) Then
          If ContInt = 0 Then
            Interval(j) = Interval(j) + 1
            STI(j, Interval(j)) = FTJ(j, k - 1)
            ContInt = 1
            IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
            - FTJ(j, k - 1) + FTJ(j, k)
            If k = Now(j) Then
              FTI(j, Interval(j)) = UpBound
              IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
            End If
          ElseIf ContInt = 1 Then
            IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
            - FTJ(j, k - 1) + FTJ(j, k)
            If k = Now(j) Then
```

```
            FTI(j, Interval(j)) = UpBound
            IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
        End If
    End If
ElseIf OrderInCrane(j, k) > Order(i) Then
    If STJ(j, k) > FTJ(j, k - 1) Then
        If ContInt = 0 Then
            Interval(j) = Interval(j) + 1
            STI(j, Interval(j)) = FTJ(j, k - 1)
            FTI(j, Interval(j)) = STJ(j, k)
            IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
        ElseIf ContInt = 1 Then
            IntTime(j, Interval(j)) = IntTime(j, Interval(j)) _
            - FTJ(j, k - 1) + STJ(j, k)
            FTI(j, Interval(j)) = STI(j, Interval(j)) + IntTime(j, Interval(j))
            ContInt = 0
        End If
        If k = Now(j) Then
            Interval(j) = Interval(j) + 1
            STI(j, Interval(j)) = FTJ(j, k)
            FTI(j, Interval(j)) = UpBound
            IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
        End If
    ElseIf STJ(j, k) = FTJ(j, k - 1) Then
        If ContInt = 1 Then
            FTI(j, Interval(j)) = STJ(j, k)
            IntTime(j, Interval(j)) = FTJ(j, k) - STJ(j, k)
            ContInt = 0
        End If
```

```
                    If k = Now(j) Then
                        Interval(j) = Interval(j) + 1
                        STI(j, Interval(j)) = FTJ(j, k)
                        FTI(j, Interval(j)) = UpBound
                        IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
                    End If
                End If
            End If
        End If
    End If
End If
Next k
End If
Next j
For j = 1 To NumofCrane
    If Interval(j) = 0 Then
        Interval(j) = 1
        STI(j, Interval(j)) = FTJ(Crane(i), Now(Crane(i)) - 1)
        FTI(j, Interval(j)) = UpBound
        IntTime(j, Interval(j)) = FTI(j, Interval(j)) - STI(j, Interval(j))
    End If
Next j
Dim STCur As Integer
Dim FTCur As Integer
Dim Ok As Integer
For j = 1 To NumofCrane
    If j <> Crane(i) Then
        For k = 1 To Interval(j)
            Ok = 0
```

```
If FTI(j, k) >= FTJ(Crane(i), Now(Crane(i)) - 1) Then
   If STI(j, k) <= FTJ(Crane(i), Now(Crane(i)) - 1) Then
      STCur = FTJ(Crane(i), Now(Crane(i)) - 1)
      FTCur = STCur + Ptime(Order(i))
   Else
      STCur = STI(j, k)
      FTCur = STCur + Ptime(Order(i))
   End If
   If FTCur <= FTI(j, k) Then
      For m = 1 To NumofCrane
         If m <> j And m <> Crane(i) Then
          Ok = 0
            For n = 1 To Interval(m)
               If STI(m, n) <= STCur And _
               FTI(m, n) >= FTCur Then
                  Ok = 1
                  Exit For
               End If
            Next n
            If Ok = 0 Then Exit For
         End If
      Next m
   End If
   If Ok = 1 Then
      STJ(Crane(i), Now(Crane(i))) = STCur
      FTJ(Crane(i), Now(Crane(i))) = STJ(Crane(i), Now(Crane(i))) _
      + Ptime(Order(i))
      Exit For
   End If
```

```
            End If
          Next k
          If Ok = 1 Then Exit For
        End If
    Next j
Next i
Dim Makespan As Integer
For i = 1 To NumofCrane
    For j = 1 To Now(i)
        If FTJ(i, j) >= Makespan Then
            Makespan = FTJ(i, j)
        End If
    Next j
Next i
Cells(39 + z, 2 + NumofJob + 2 + NumofJob + 1) = Makespan
Cells(39 + z, 2 + NumofJob + 2 + NumofJob + 1).Interior.Color = RGB(250, 150, 80)
Cells(39 + z, 2 + NumofJob + 2 + NumofJob + 1).Borders.Weight = xlThin
End Sub
```

ตารางที่ ข-16 แสดงโค้ดคำสั่งที่สั่งให้ทำการสุ่มค่าต้นแบบใหม่อยู่ในส่วนการประมวลผล

```
Sub Newns1()
NumofJob = Worksheets("EnterData").Cells(8, 5)
NumofCrane = Worksheets("EnterData").Cells(9, 5)
 Dim iArr() As Integer
      ReDim iArr(1 To NumofJob)
      For i = 1 To NumofJob
            iArr(i) = i
      Next i
```

```
    For i = NumofJob To 1 Step -1
        R = Int(Rnd() * (i - 1)) + 1
        Temp = iArr(R)
        iArr(R) = iArr(i)
        iArr(i) = Temp
    Next i
    For i = 1 To NumofJob
      Worksheets("Solve").Cells(23, 2 + i) = iArr(i)
      Worksheets("Solve").Cells(23, 2 + i).Interior.Color = RGB(186, 250, 211)
      Worksheets("Solve").Cells(23, 2 + i).Borders.Weight = xlThin
    Next i
    For i = 1 To NumofJob
      Worksheets("Solve").Cells(24, 2 + i) = 0
    Next i
        R1 = Int((NumofJob - 1) * Rnd() + 1)
        Worksheets("Solve").Cells(24, 2 + R1) = 1
      For i = 2 To NumofCrane
        Do
          R1 = Int((NumofJob - 1) * Rnd() + 1)
        Loop Until Worksheets("Solve").Cells(24, 2 + R1) = 0
        Worksheets("Solve").Cells(24, 2 + R1) = i
      Next i
      For i = 1 To NumofJob
        If Worksheets("Solve").Cells(24, 2 + i) = 0 Then
            Worksheets("Solve").Cells(24, 2 + i) = Int((NumofCrane * Rnd()) + 1)
        End If
      Next i
End Sub
```