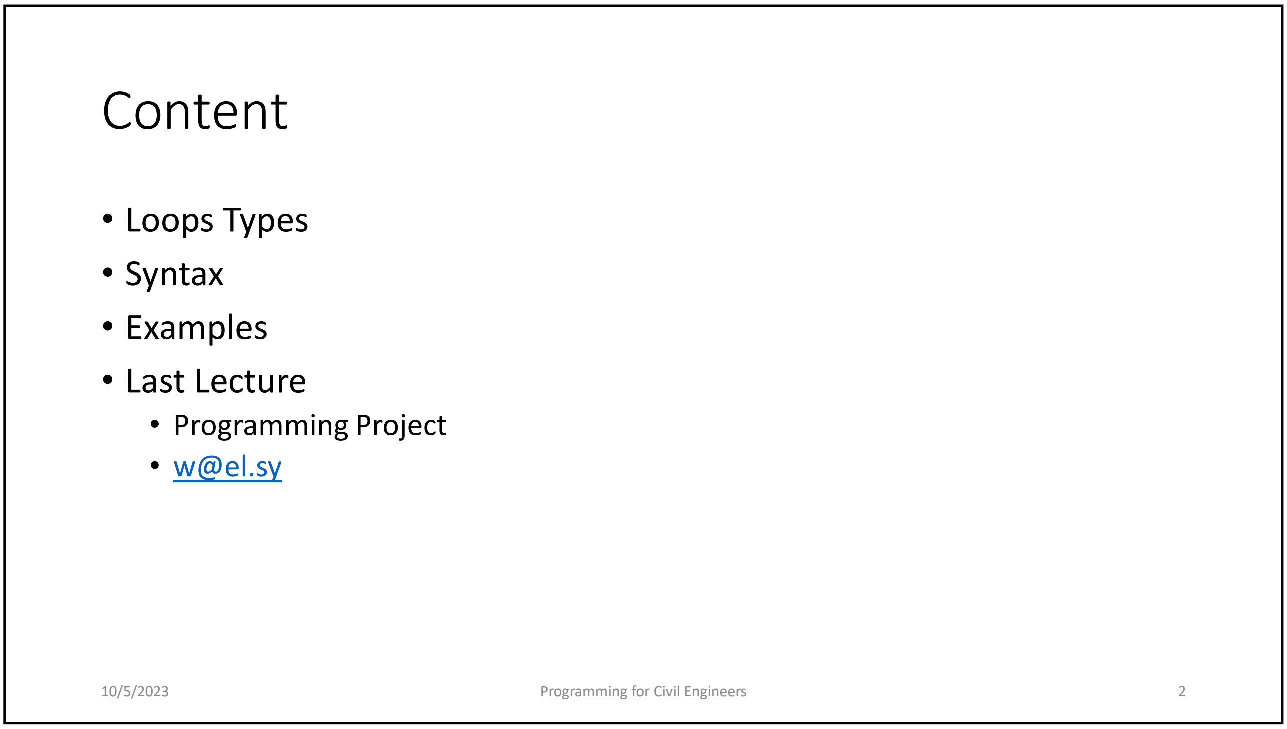
جامعة دمشق
كلية الهندسة المدنية

مادة البرمجة
السنة الثالثة
المحاضرة 5

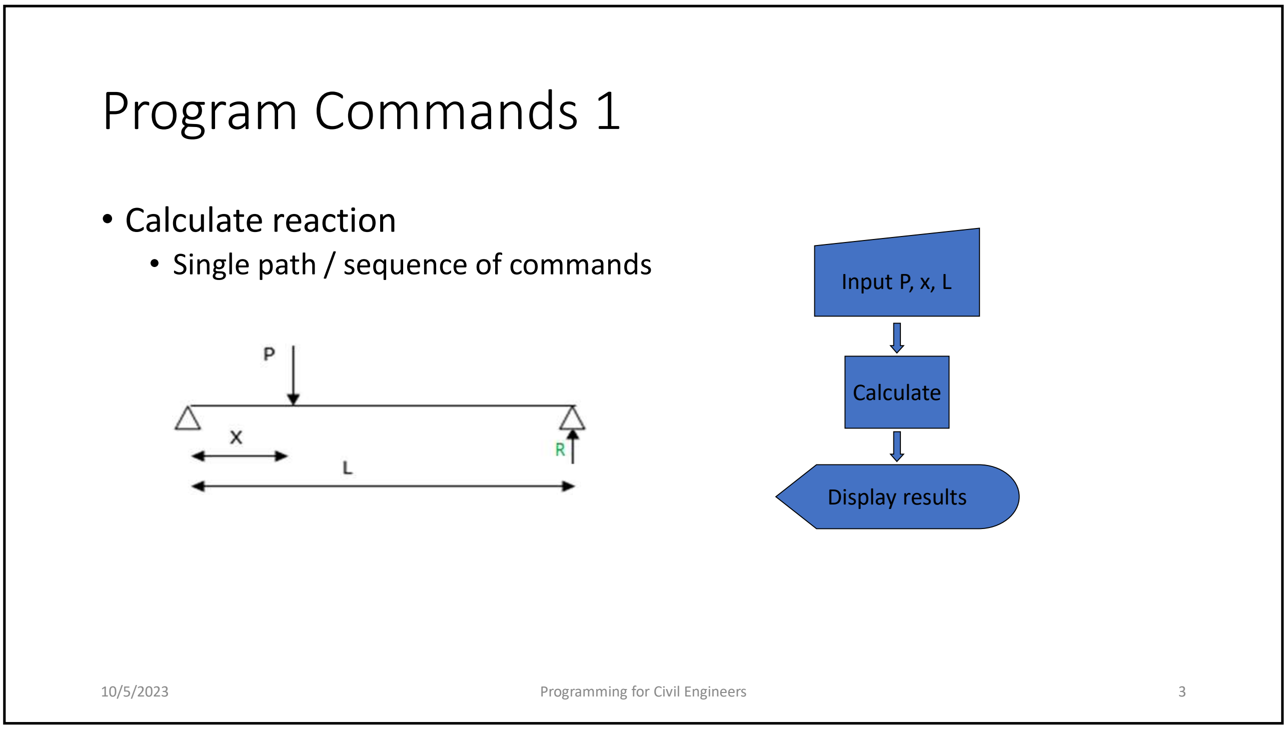
10/5/2023

Wael Darwich



Content

- Loops Types
- Syntax
- Examples
- Last Lecture
  - Programming Project
  - w@el.sy



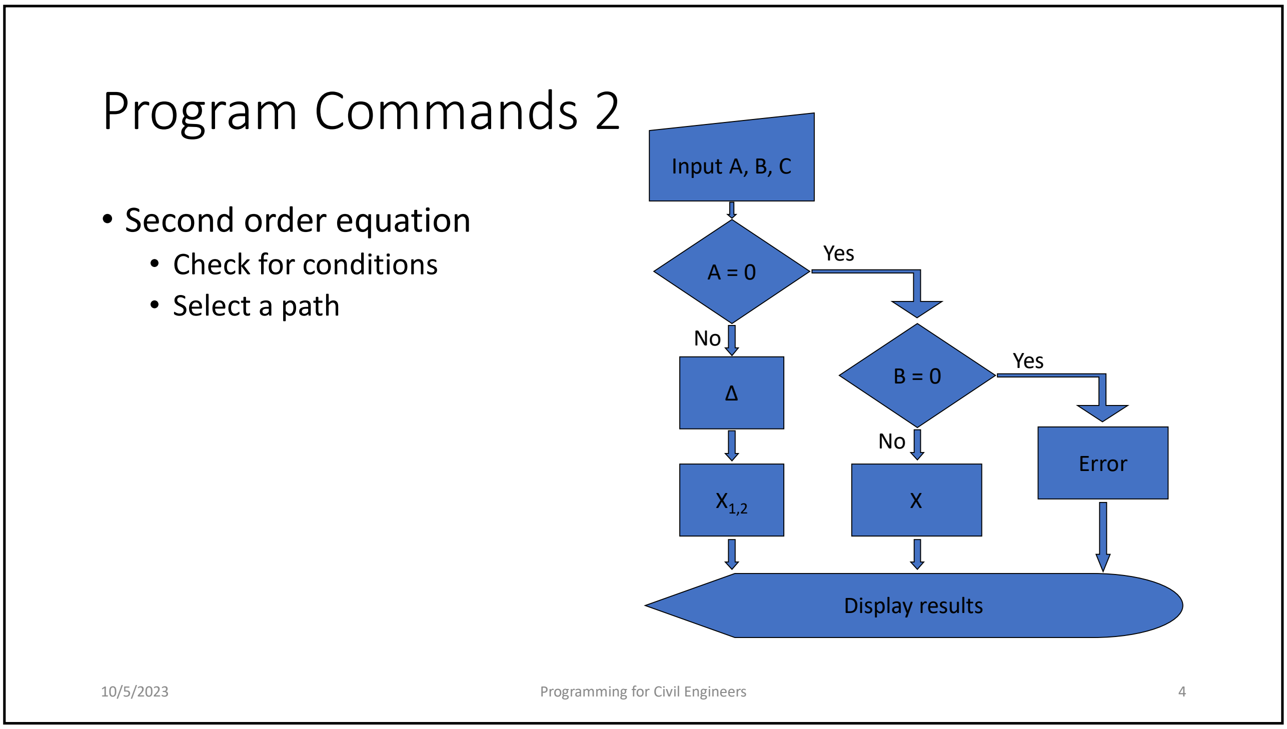
Program Commands 1

- Calculate reaction
  - Single path / sequence of commands

Input P, x, L

Calculate

Display results



Program Commands 2

- Second order equation
  - Check for conditions
  - Select a path

Input A, B, C

A = 0

Yes

No

Δ

$X_{1,2}$

B = 0

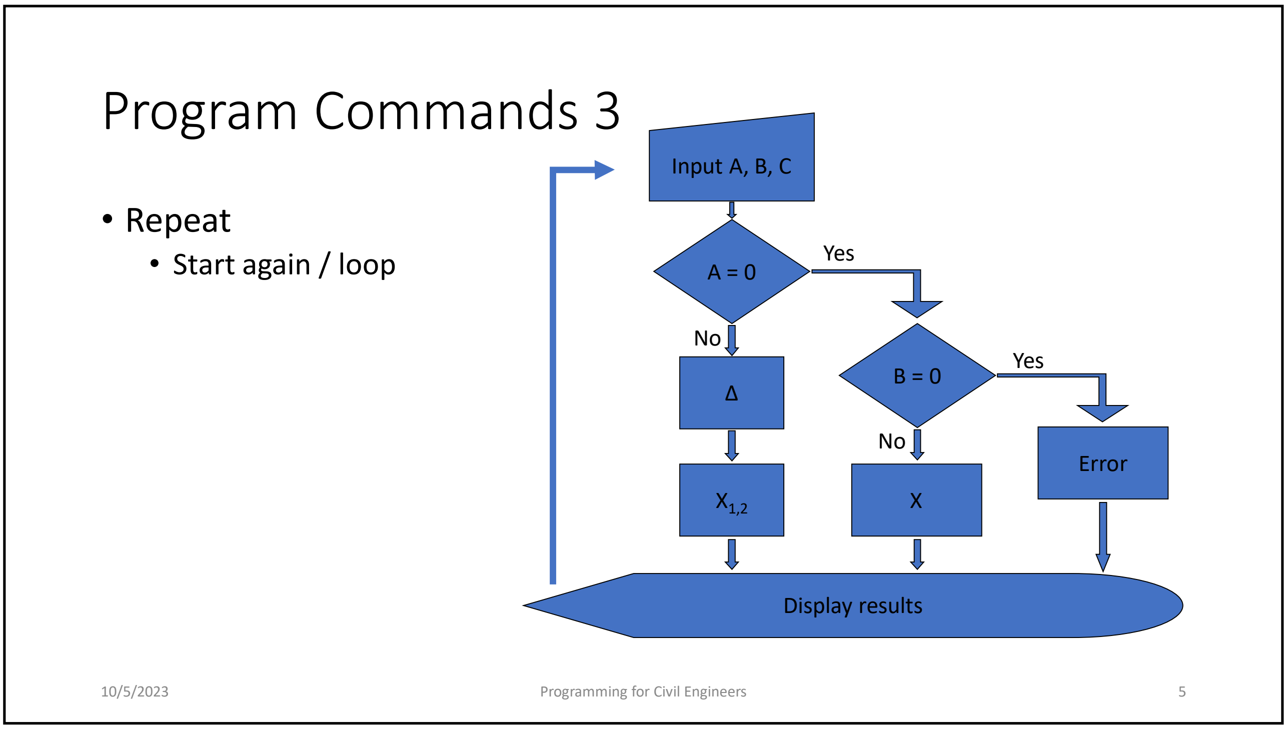
Yes

No

X

Error

Display results



Program Commands 3

- Repeat
  - Start again / loop

Input A, B, C

A = 0

Yes

No

B = 0

Yes

No

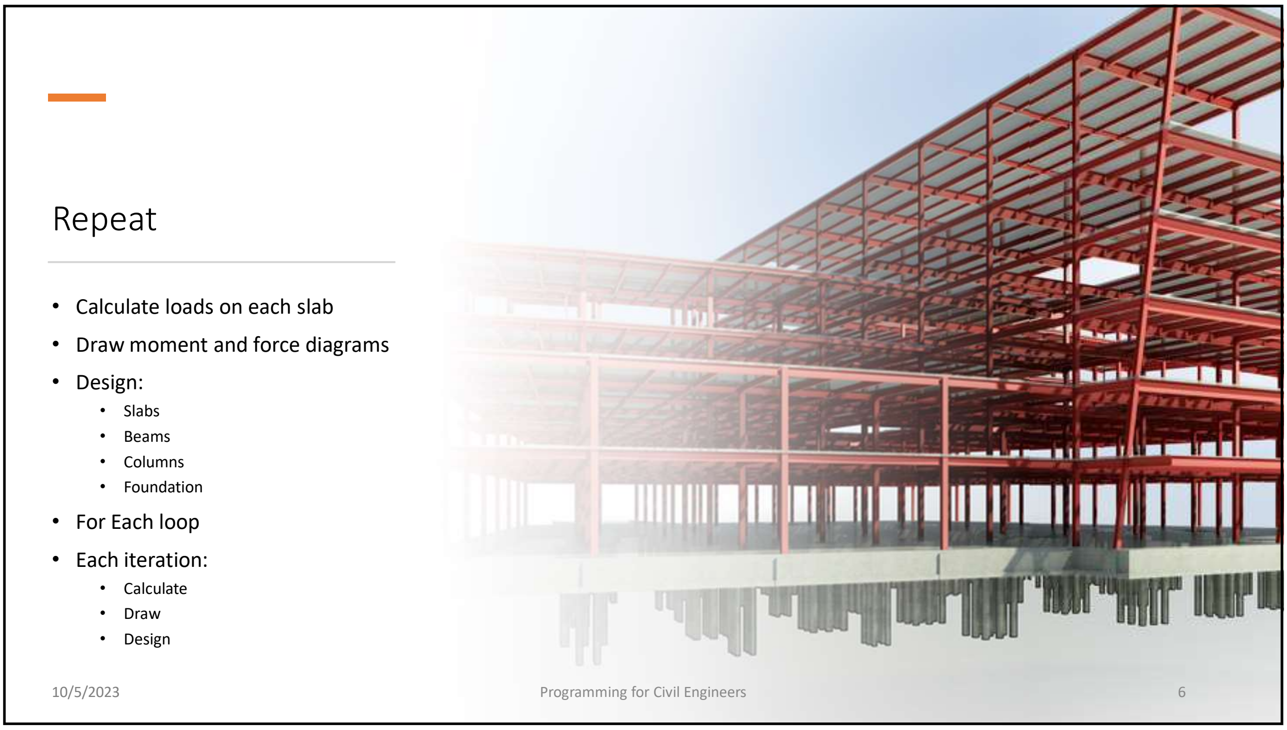
Δ

$X_{1,2}$

X

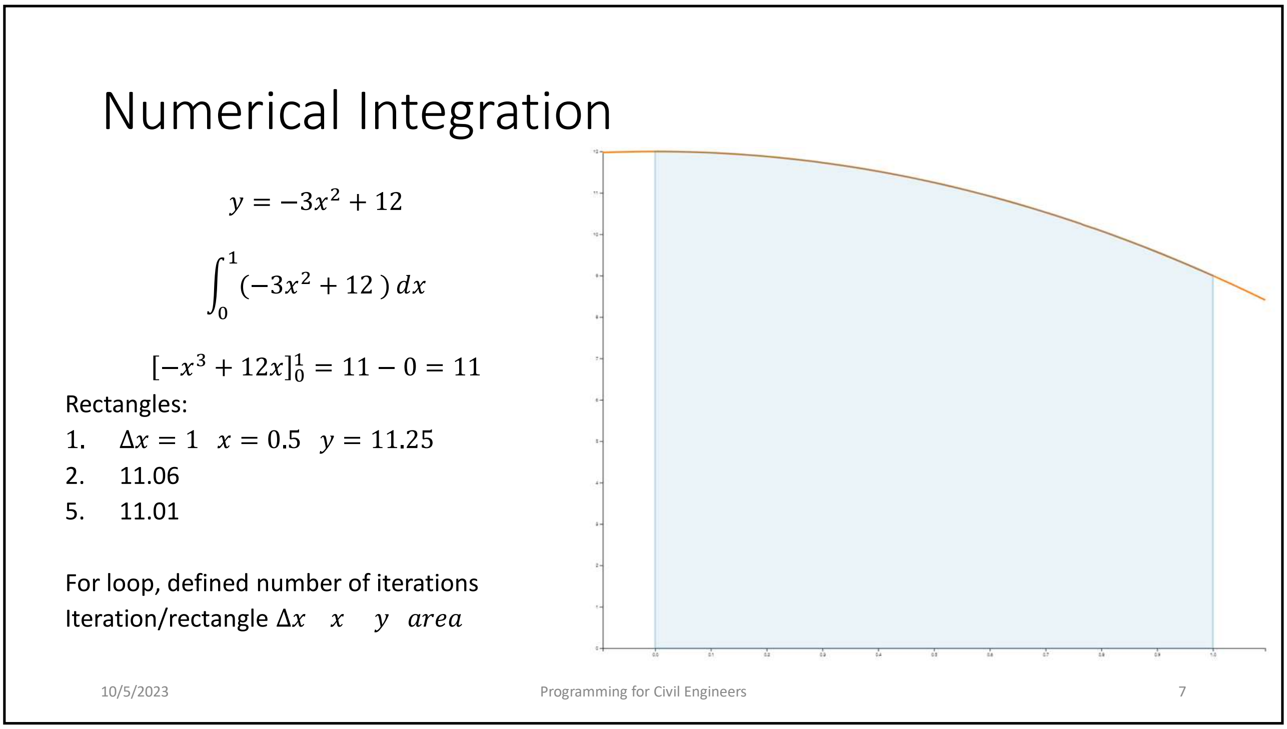
Error

Display results



Repeat

- Calculate loads on each slab
- Draw moment and force diagrams
- Design:
  - Slabs
  - Beams
  - Columns
  - Foundation
- For Each loop
- Each iteration:
  - Calculate
  - Draw
  - Design



Numerical Integration

$$y = -3x^2 + 12$$

$$\int_0^1 (-3x^2 + 12\,)\,dx$$

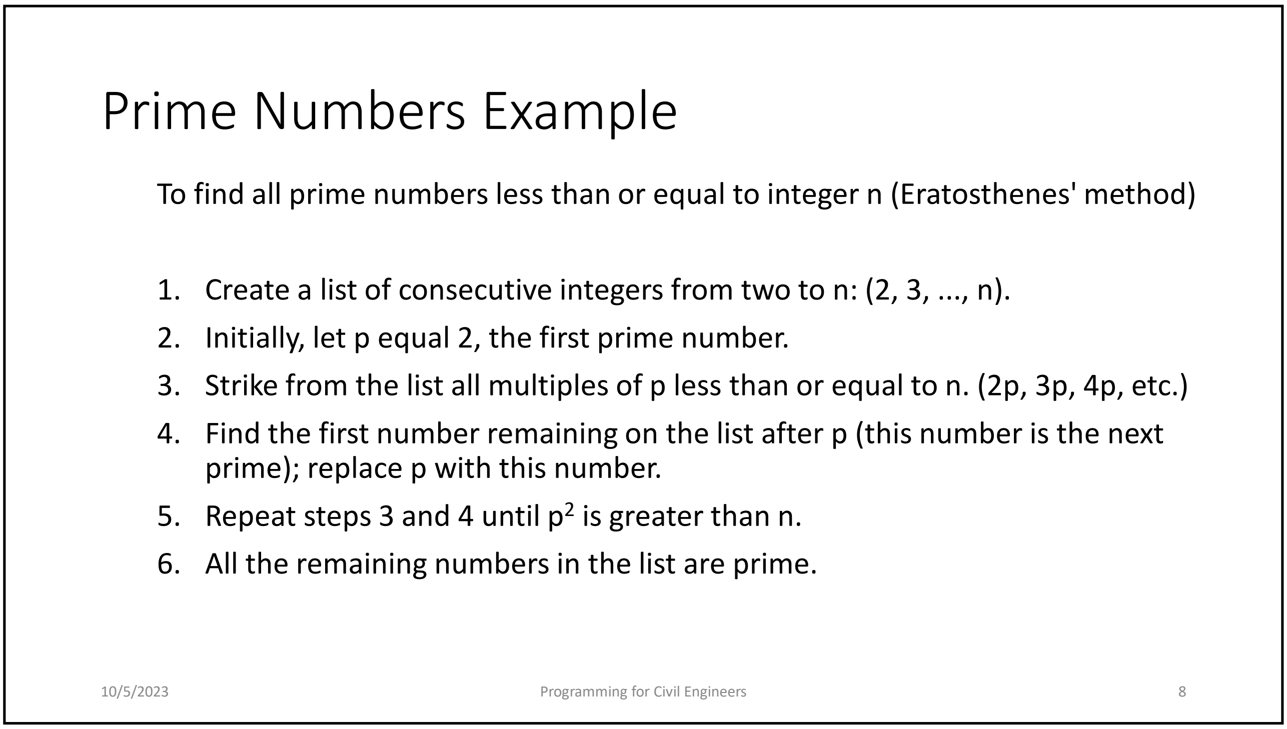
$$[-x^3 + 12x]_0^1 = 11 - 0 = 11$$

Rectangles:

1. $\Delta x = 1 \quad x = 0.5 \quad y = 11.25$
2. 11.06
5. 11.01

For loop, defined number of iterations

Iteration/rectangle $\Delta x \quad x \quad y \quad area$

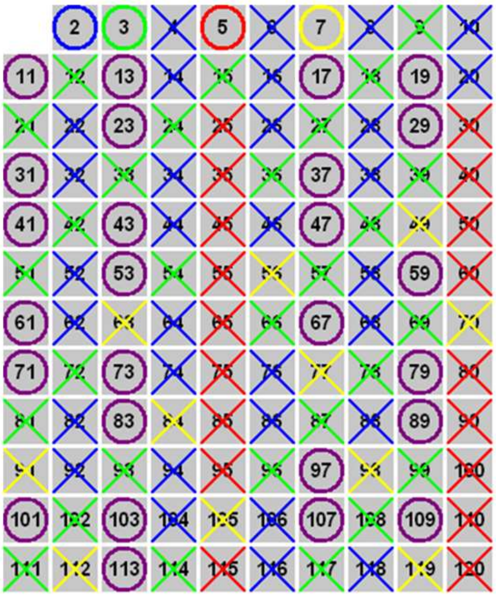


Prime Numbers Example

To find all prime numbers less than or equal to integer n (Eratosthenes' method)

1. Create a list of consecutive integers from two to n: (2, 3, ..., n).
2. Initially, let p equal 2, the first prime number.
3. Strike from the list all multiples of p less than or equal to n. (2p, 3p, 4p, etc.)
4. Find the first number remaining on the list after p (this number is the next prime); replace p with this number.
5. Repeat steps 3 and 4 until p^2 is greater than n.
6. All the remaining numbers in the list are prime.

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120



Prime numbers

2	3	5	7
11	13	17	19
23	29	31	37
41	43	47	53
59	61	67	71
73	79	83	89
97	101	103	107
109	113		

Series Examples

- Cosine and Sine
- Angle in Radian
- Finite vs infinite series
  - Accuracy
  - 10!=3.6E6
  - 20!=2.4E18

$$\cos(x)=1-\frac{x^2}{2!}+\frac{x^4}{4!}-\frac{x^6}{6!}+\dots$$

n: 0 1 2 3

2n: 0 2 4 6

±: -1^0 -1^1 -1^2 -1^3

$$=\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n)!}x^{2n}$$

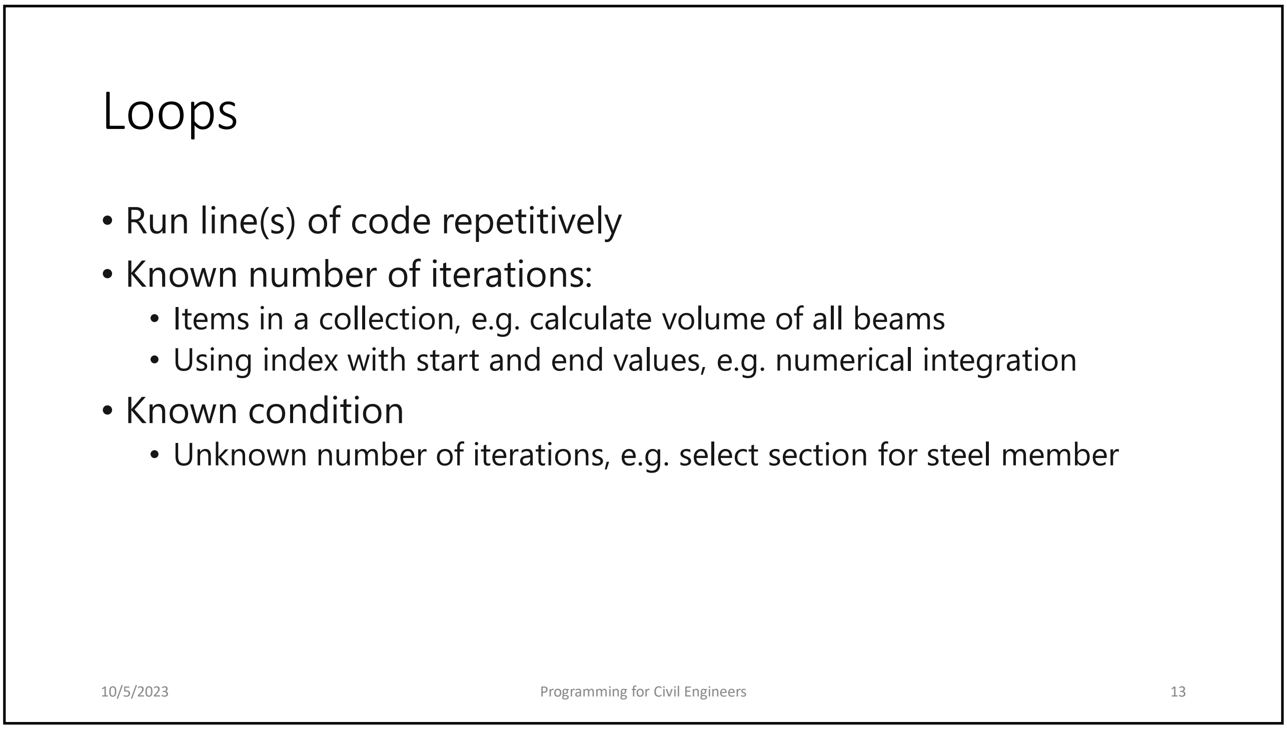
$$\sin(x)=x-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}+\dots$$

1 3 5 7

$$=\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n+1)!}x^{2n+1}$$

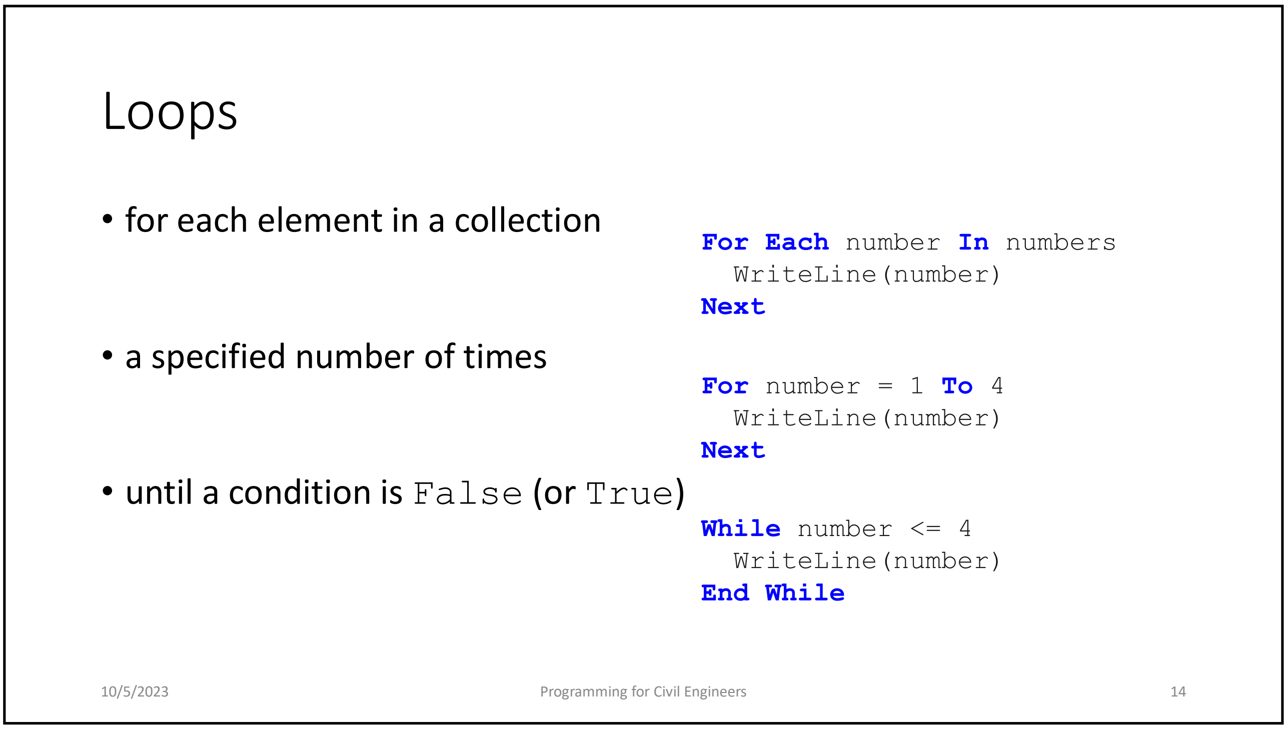
Repeat

- Until a condition is satisfied
- While loop
- Unknown number of iterations
- Check condition



Loops

- Run line(s) of code repetitively
- Known number of iterations:
  - Items in a collection, e.g. calculate volume of all beams
  - Using index with start and end values, e.g. numerical integration
- Known condition
  - Unknown number of iterations, e.g. select section for steel member



Loops

- for each element in a collection

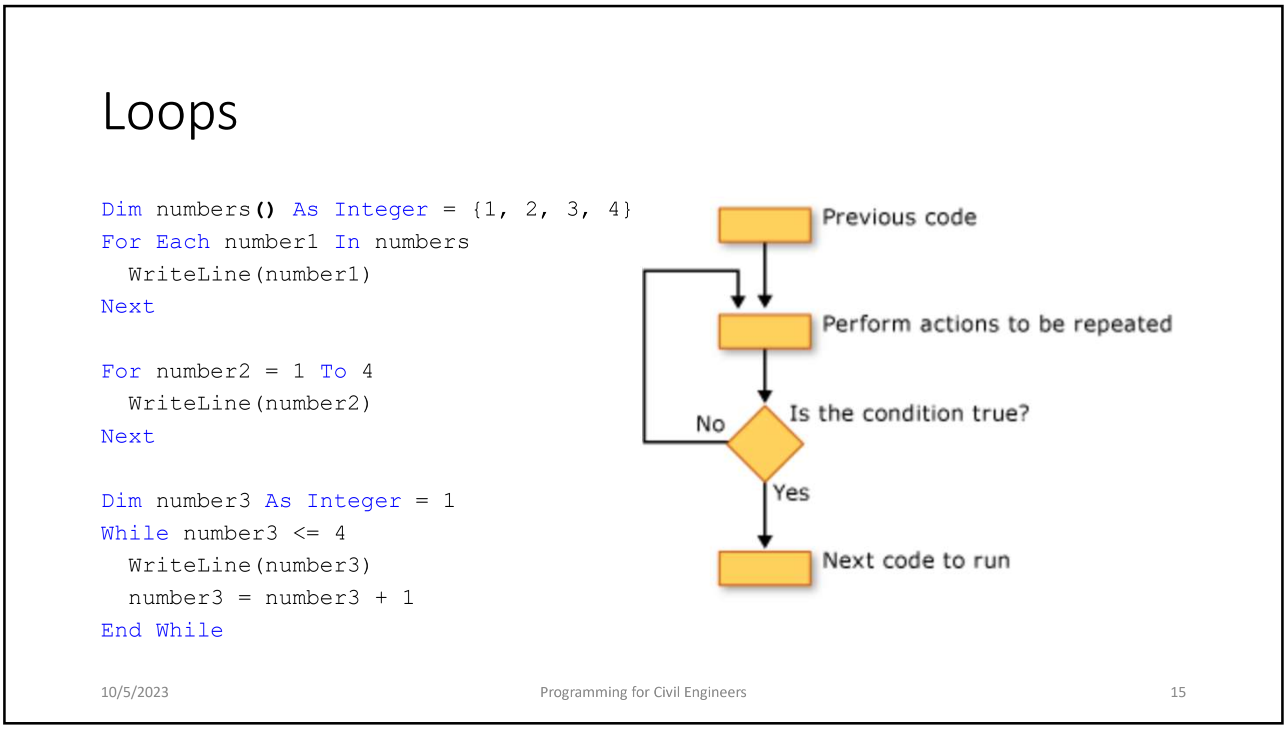
```
For Each number In numbers
  WriteLine(number)
Next
```

- a specified number of times

```
For number = 1 To 4
  WriteLine(number)
Next
```

- until a condition is `False` (or `True`)

```
While number <= 4
  WriteLine(number)
End While
```

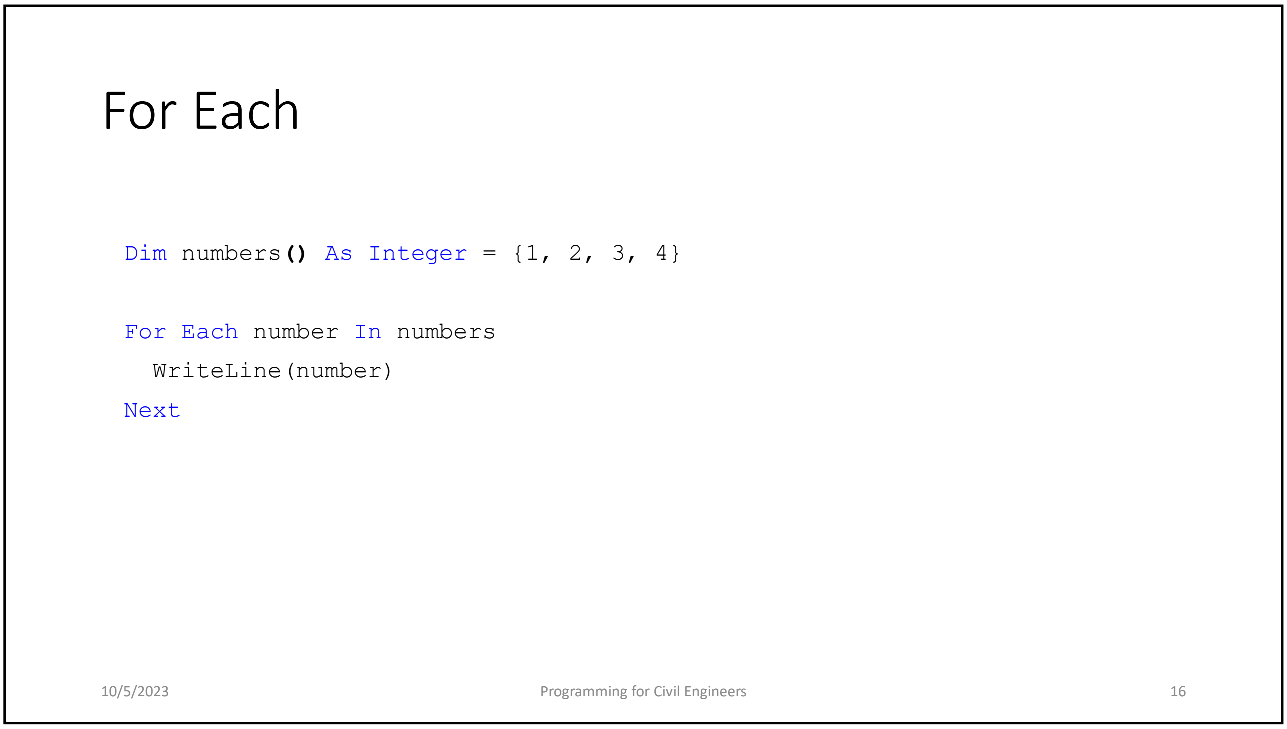


Loops

```
Dim numbers() As Integer = {1, 2, 3, 4}
For Each number1 In numbers
  WriteLine(number1)
Next

For number2 = 1 To 4
  WriteLine(number2)
Next

Dim number3 As Integer = 1
While number3 <= 4
  WriteLine(number3)
  number3 = number3 + 1
End While
```

Previous code
Perform actions to be repeated
No
Is the condition true?
Yes
Next code to run

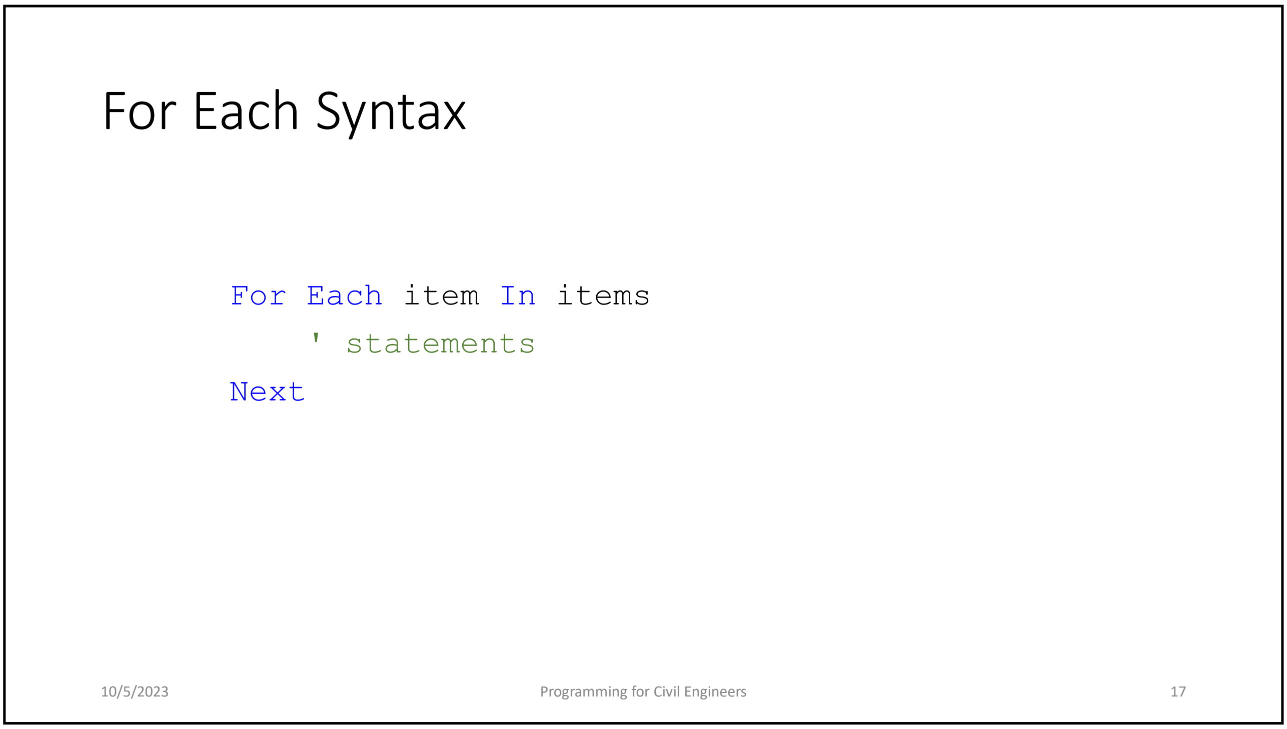
15



For Each

```
Dim numbers() As Integer = {1, 2, 3, 4}

For Each number In numbers
  WriteLine(number)
Next
```

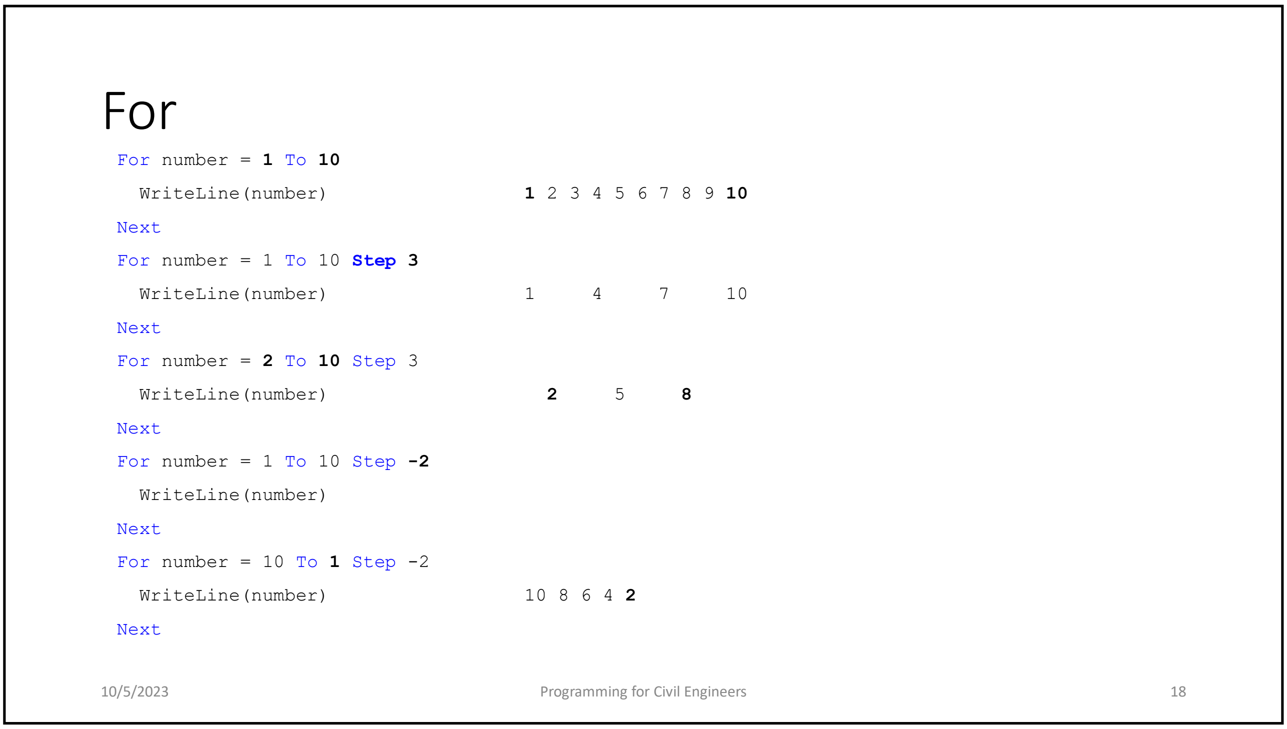
16



For Each Syntax

```
For Each item In items
    ' statements
Next
```

17



For

```
For number = 1 To 10
  WriteLine(number)                1 2 3 4 5 6 7 8 9 10
Next
For number = 1 To 10 Step 3
  WriteLine(number)                1     4     7     10
Next
For number = 2 To 10 Step 3
  WriteLine(number)                  2     5     8
Next
For number = 1 To 10 Step -2
  WriteLine(number)
Next
For number = 10 To 1 Step -2
  WriteLine(number)                10 8 6 4 2
Next
```

18

For Syntax

Index i   Or use start, end and step   1st   **Could** be last

```
For counter [ As datatype ] = start To end [ Step step ]
    ' statements
Next [ counter ]
```

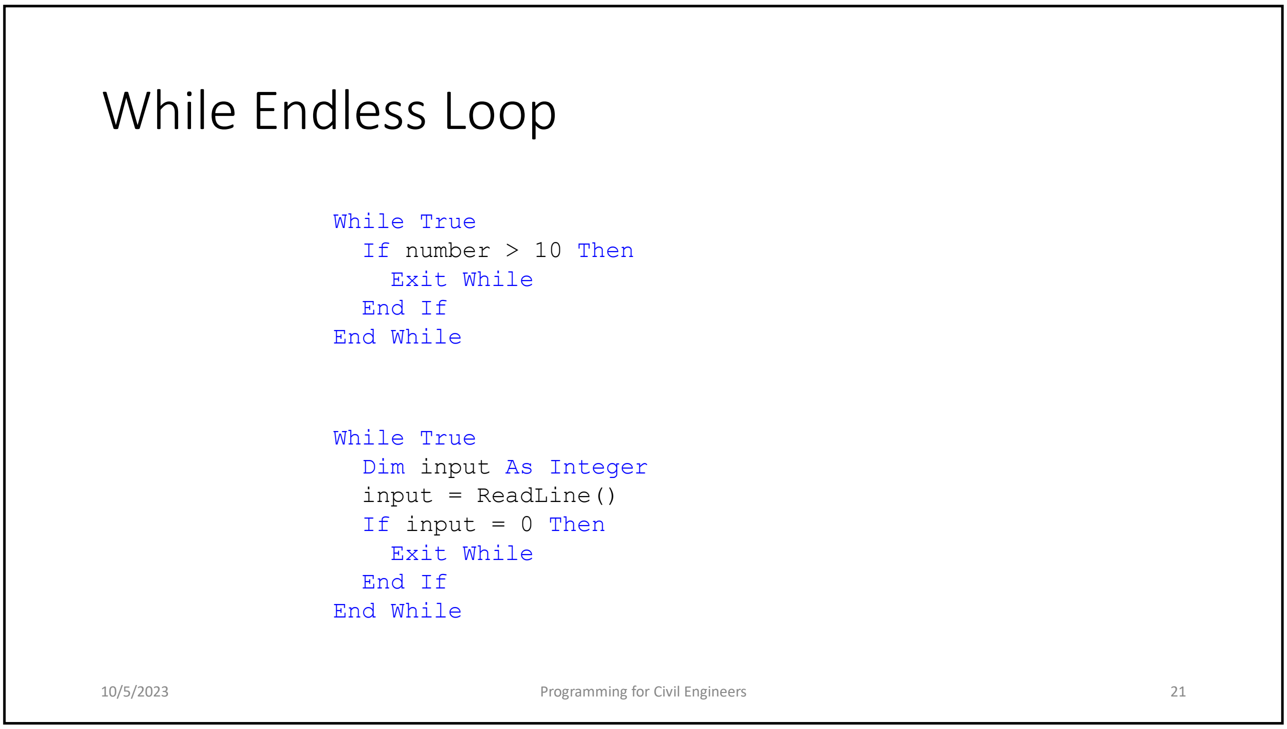
Default = 1

unlike `' counter`

While

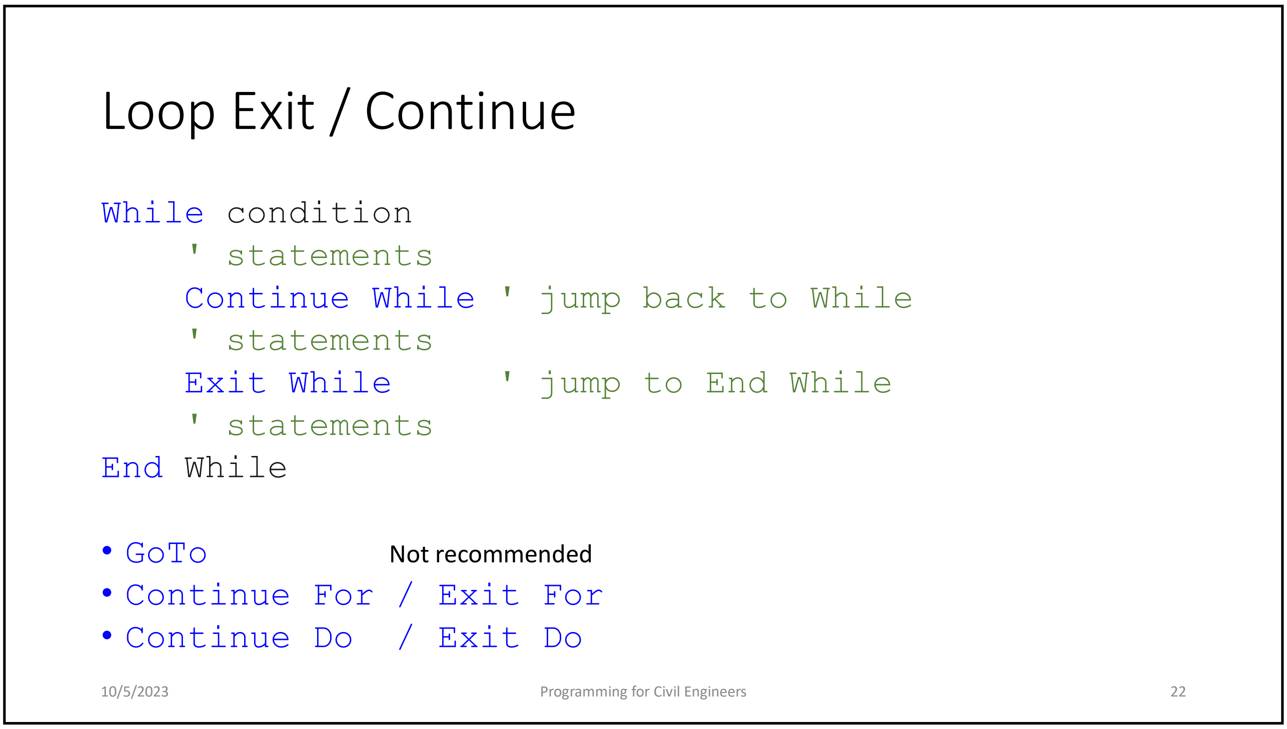
```
code1
While number <= 4
  code2
  number = number + 1
End While
```

- Use condition instead of counter
- Unknown number of iterations
- Endless loop
- `code1` : Needs to satisfy the condition to get in
- `code2` : Needs to break the condition to get out



While Endless Loop

```
While True
  If number > 10 Then
    Exit While
  End If
End While


While True
  Dim input As Integer
  input = ReadLine()
  If input = 0 Then
    Exit While
  End If
End While
```



Loop Exit / Continue

```
While condition
     ' statements
     Continue While ' jump back to While
     ' statements
     Exit While     ' jump to End While
     ' statements
End While
```

- GoTo Not recommended
- Continue For / Exit For
- Continue Do / Exit Do

Do

```
While condition
 ' statements
End While
```

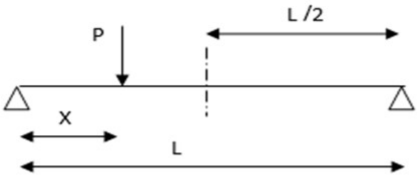
```
Do
 ' statements
Loop While condition

Do While condition
 ' statements
Loop
```

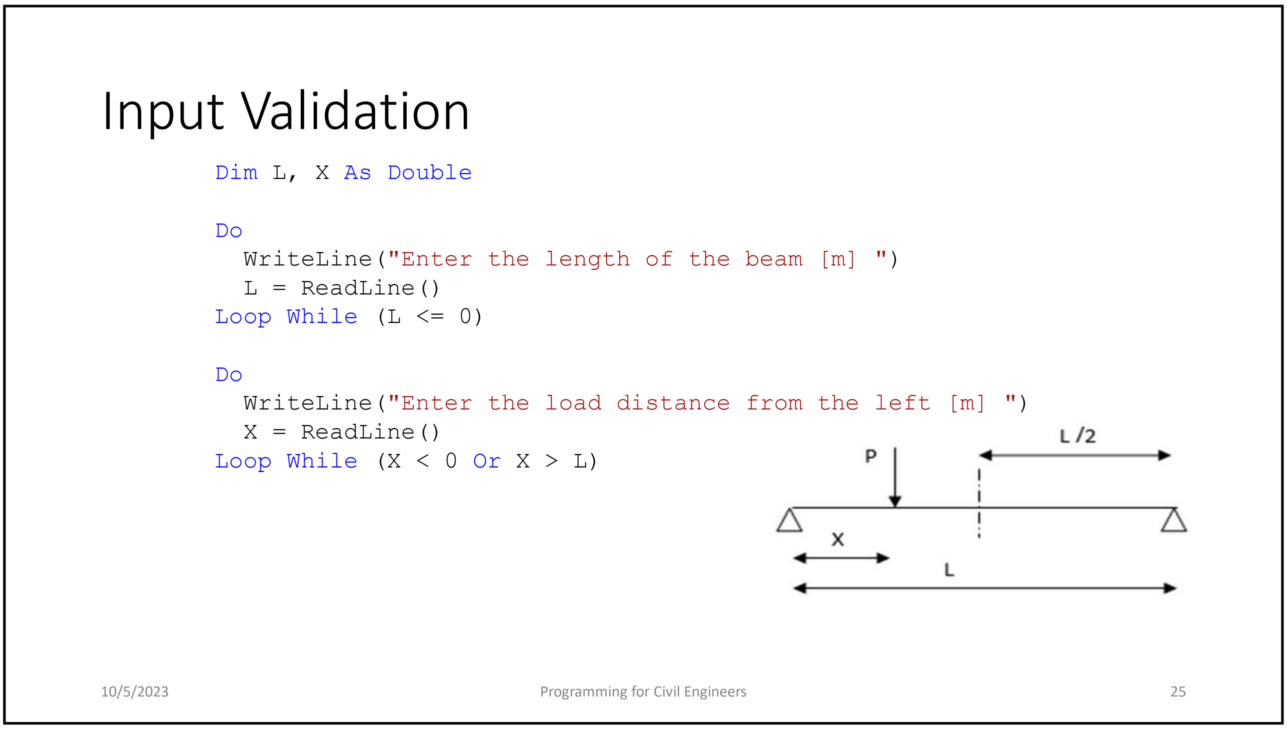
```
Do
 ' statements
Loop Until condition

Do Until condition
 ' statements
Loop
```

- Check condition at the beginning or end of the iteration
- Check for `True` or `False`
  - `While condition`
  - `Until Not condition`

Input Validation

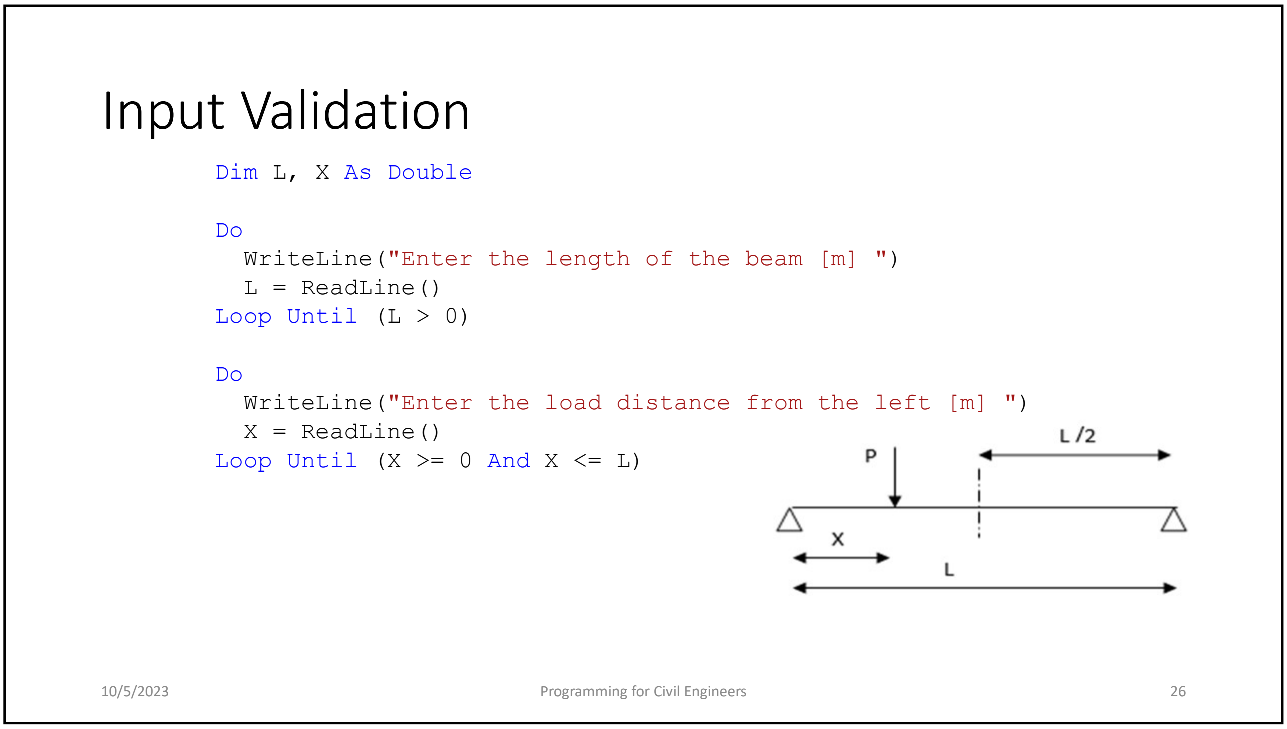
```
Dim L, X As Double

WriteLine("Enter the length of the beam [m] ")
L = ReadLine()
WriteLine("Enter the load distance from the left [m] ")
X = ReadLine()

If (L <= 0 Or X < 0 Or X > L) Then
  WriteLine("Input Error!")
Else
  ' Calculate...
End If
```





Input Validation

```
Dim L, X As Double

Do
  WriteLine("Enter the length of the beam [m] ")
  L = ReadLine()
Loop While (L <= 0)

Do
  WriteLine("Enter the load distance from the left [m] ")
  X = ReadLine()
Loop While (X < 0 Or X > L)
```



Input Validation

```
Dim L, X As Double

Do
  WriteLine("Enter the length of the beam [m] ")
  L = ReadLine()
Loop Until (L > 0)

Do
  WriteLine("Enter the load distance from the left [m] ")
  X = ReadLine()
Loop Until (X >= 0 And X <= L)
```

Loops Syntax

```
For counter = start To end [ Step step ]
    ' statements
Next [ counter ]
```

```
For Each item In items
    ' statements
Next
```

```
While condition
   ' statements
End While
```

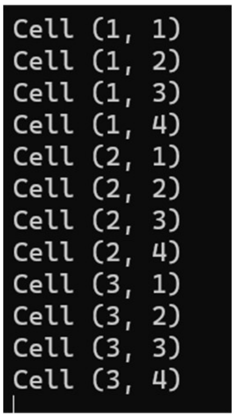
```
Do [ { While | Until } condition ]
   ' statements
Loop
```

```
Do
   ' statements
Loop [ { While | Until } condition ]
```

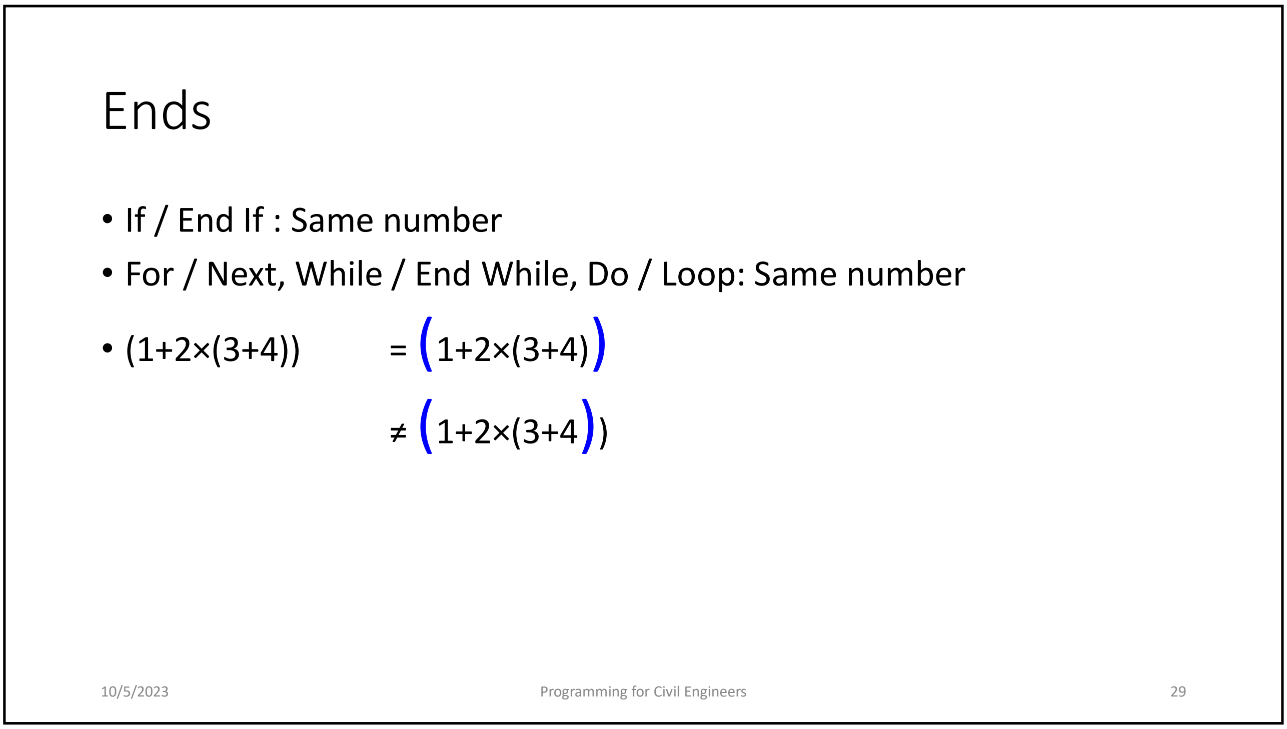
Nested Loops

	1	**2**	**3**	**4**
1	1,1	1,2	1,3	1,4
2	2,1	2,2	2,3	2,4
3	3,1	3,2	3,3	3,4

```
For row = 1 To 3
  For column = 1 To 4
    WriteLine("Cell (" & row & ", " & column & ")")
  Next column
Next row
```



```
Cell (1, 1)
Cell (1, 2)
Cell (1, 3)
Cell (1, 4)
Cell (2, 1)
Cell (2, 2)
Cell (2, 3)
Cell (2, 4)
Cell (3, 1)
Cell (3, 2)
Cell (3, 3)
Cell (3, 4)
```

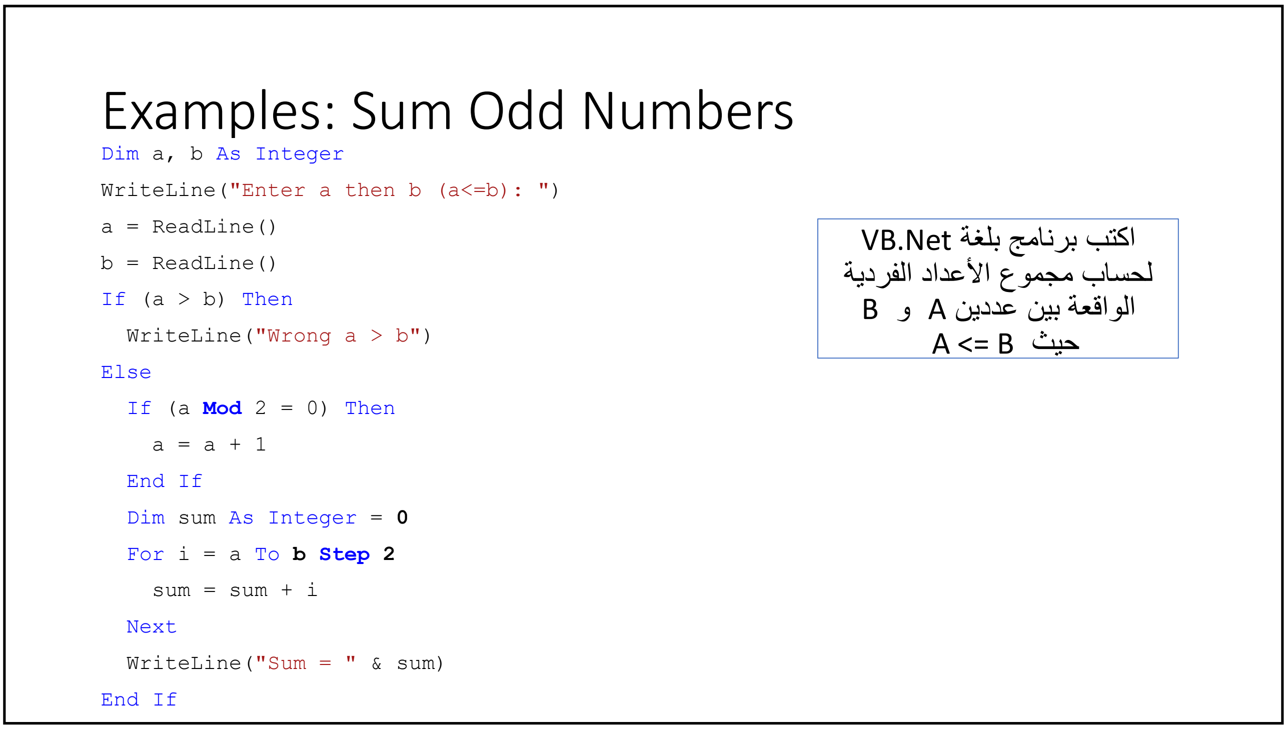


Ends

- If / End If : Same number
- For / Next, While / End While, Do / Loop: Same number
- (1+2×(3+4)) = (1+2×(3+4))

  ≠ (1+2×(3+4))

29

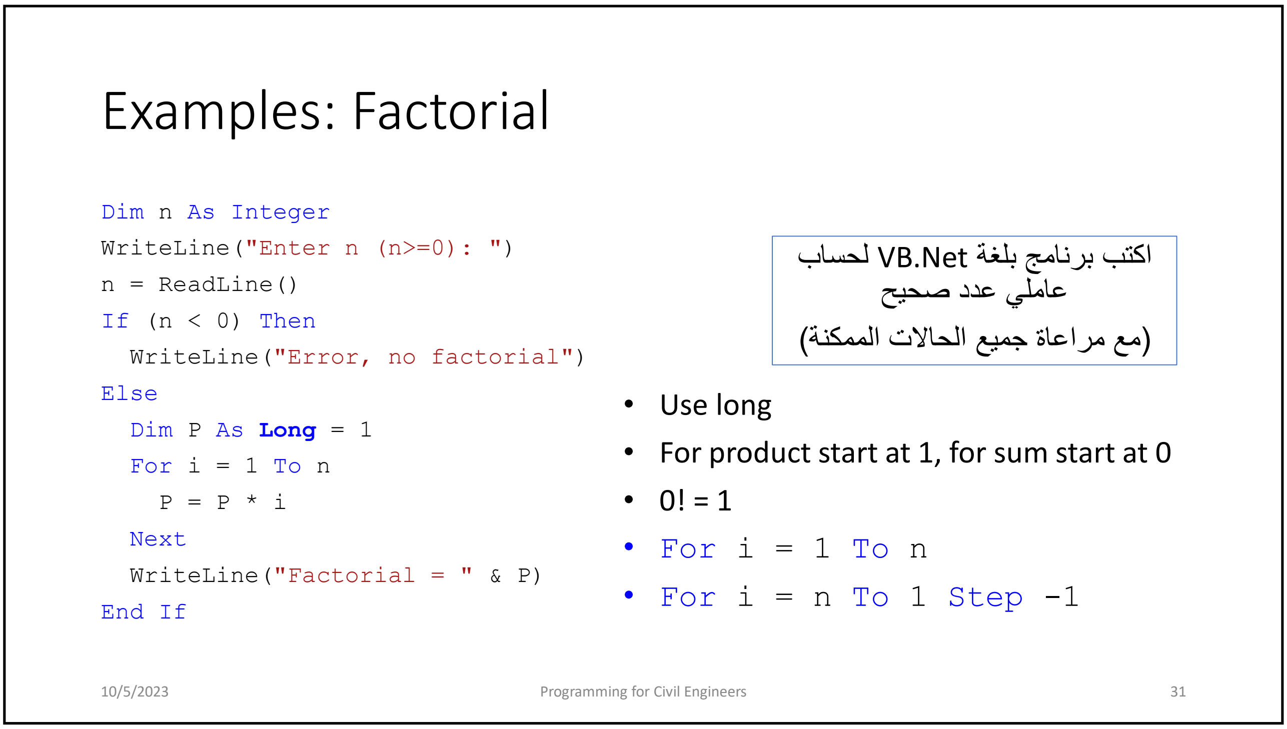


Examples: Sum Odd Numbers

```
Dim a, b As Integer
WriteLine("Enter a then b (a<=b): ")
a = ReadLine()
b = ReadLine()
If (a > b) Then
  WriteLine("Wrong a > b")
Else
  If (a Mod 2 = 0) Then
    a = a + 1
  End If
  Dim sum As Integer = 0
  For i = a To b Step 2
    sum = sum + i
  Next
  WriteLine("Sum = " & sum)
End If
```

اكتب برنامج بلغة VB.Net
لحساب مجموع الأعداد الفردية
الواقعة بين عددين A و B
حيث A <= B

30

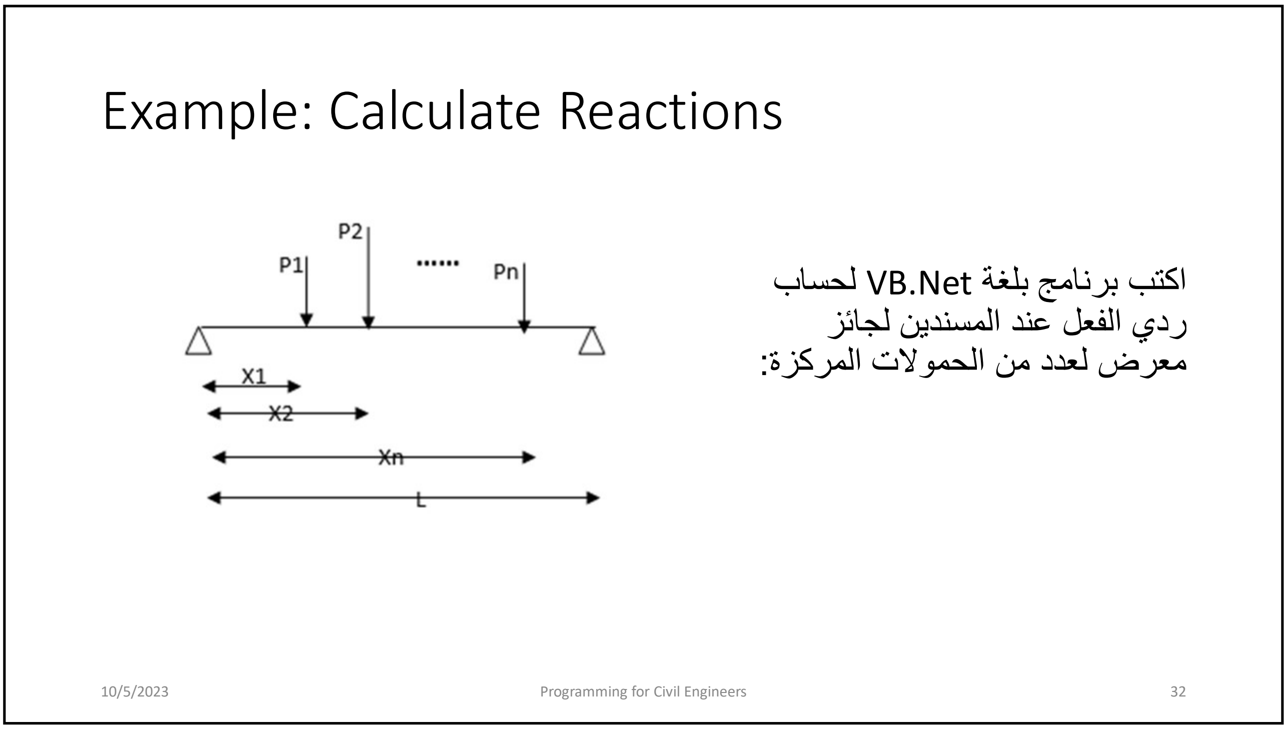


Examples: Factorial

```
Dim n As Integer
WriteLine("Enter n (n>=0): ")
n = ReadLine()
If (n < 0) Then
  WriteLine("Error, no factorial")
Else
  Dim P As Long = 1
  For i = 1 To n
    P = P * i
  Next
  WriteLine("Factorial = " & P)
End If
```

اكتب برنامج بلغة VB.Net لحساب
عاملي عدد صحيح
(مع مراعاة جميع الحالات الممكنة)

- Use long
- For product start at 1, for sum start at 0
- 0! = 1
- `For i = 1 To n`
- `For i = n To 1 Step -1`



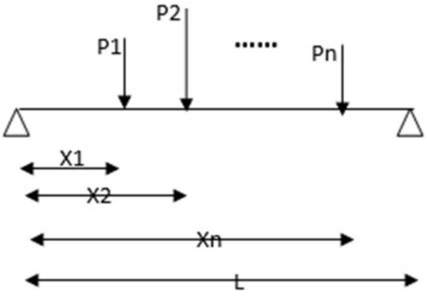
Example: Calculate Reactions

اكتب برنامج بلغة VB.Net لحساب
ردي الفعل عند المسندين لجائز
معرض لعدد من الحمولات المركزة:

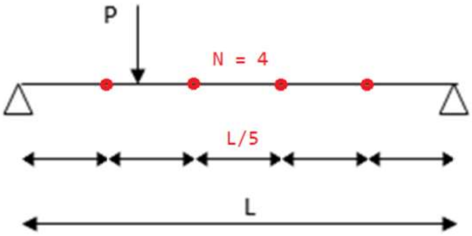
Example: Calculate Reactions

```
WriteLine("Enter beam length: ")
Dim L As Double = ReadLine()

WriteLine("Enter number of loads: ")
Dim N As Integer = ReadLine()

If (L <= 0 Or N < 0) Then
  WriteLine("Wrong Input!")
Else

End If
```

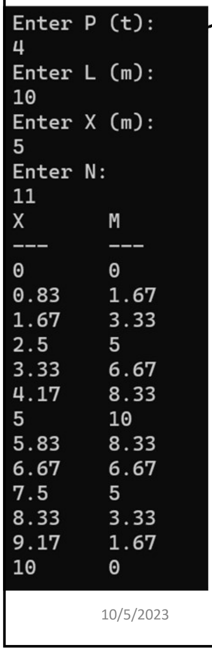
```
Dim P, X, SigmaM, SigmaY, RR, RL As Double
SigmaM = 0
SigmaY = 0
For i = 1 To N
  WriteLine("Enter P then X for Load " & i)
  P = ReadLine()
  X = ReadLine()
  If (X < 0 Or X > L) Then
    WriteLine("Wrong entry, it will be ignored!")
  Else
    SigmaY = SigmaY + P
    SigmaM = SigmaM + P * X
  End If
Next
RR = SigmaM / L
RL = SigmaY - RR
WriteLine("RR = " & RR)
WriteLine("RL = " & RL)
```



Example: Calculate Bending Moment



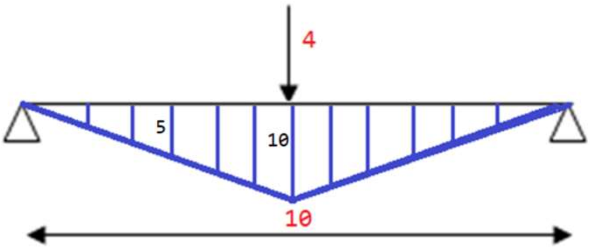
اكتب برنامج بلغة VB.Net لحساب عزم
الانعطاف في عدد من النقاط الموزعة N
على طول الجائز تحت تأثير حمولة
مركزة و طباعة النتائج في جدول:

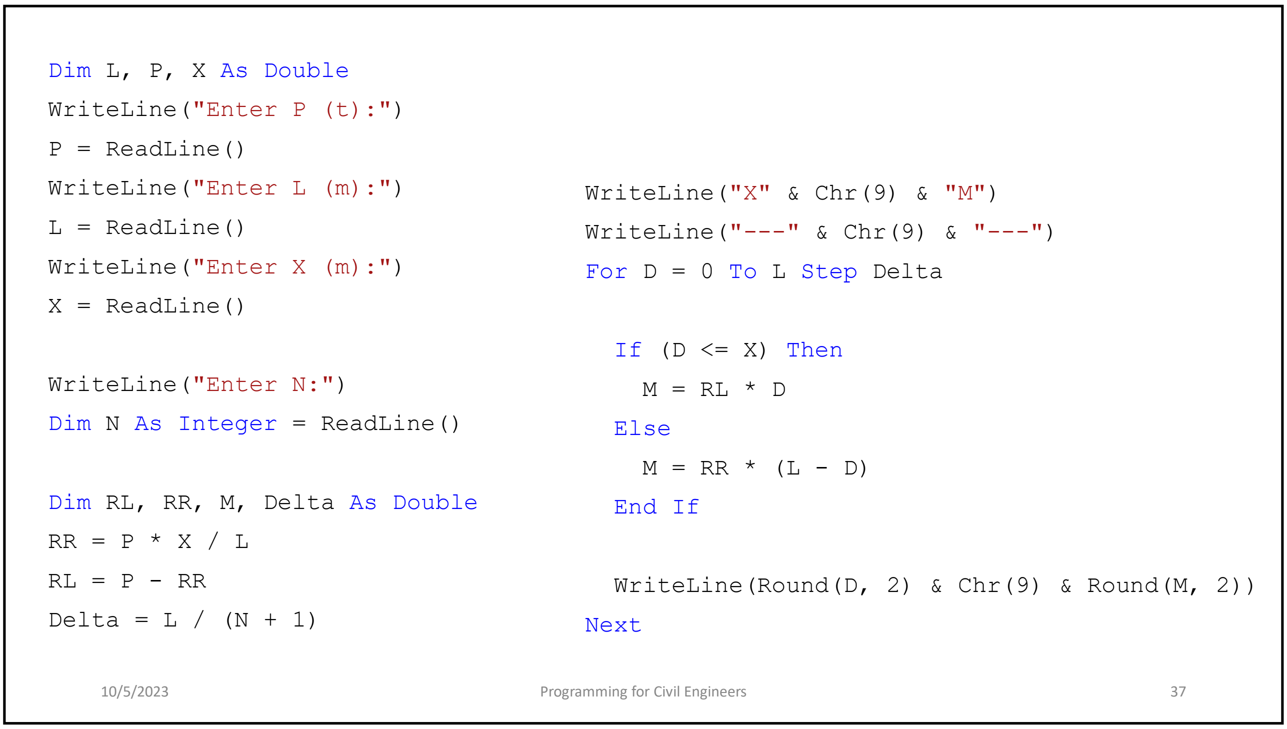


```
Enter P (t):
4
Enter L (m):
10
Enter X (m):
5
Enter N:
11
X       M
---     ---
0       0
0.83    1.67
1.67    3.33
2.5     5
3.33    6.67
4.17    8.33
5       10
5.83    8.33
6.67    6.67
7.5     5
8.33    3.33
9.17    1.67
10      0
```

nple: Calculate Bending Moment

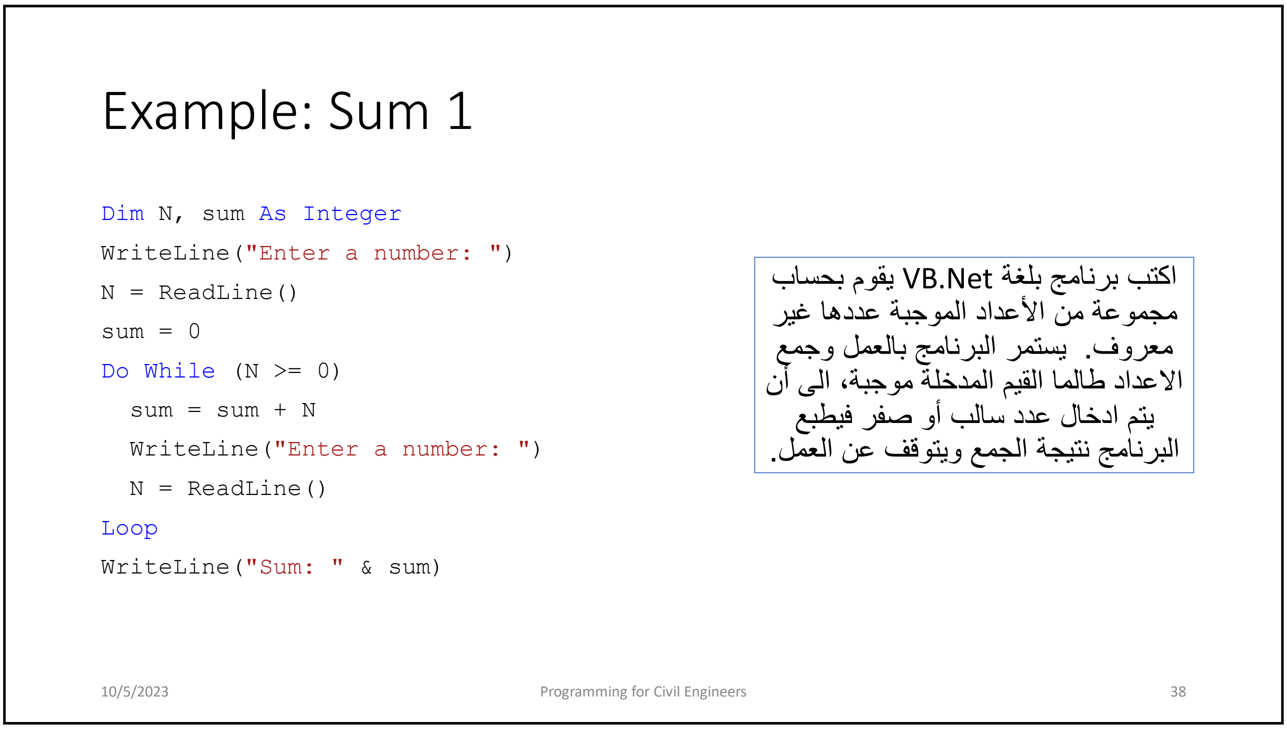
اكتب برنامج بلغة VB.Net لحساب عزم
الانعطاف في عدد من النقاط الموزعة N
على طول الجائز تحت تأثير حمولة
مركزة و طباعة النتائج في جدول:





```
Dim L, P, X As Double
WriteLine("Enter P (t):")
P = ReadLine()
WriteLine("Enter L (m):")
L = ReadLine()
WriteLine("Enter X (m):")
X = ReadLine()

WriteLine("Enter N:")
Dim N As Integer = ReadLine()

Dim RL, RR, M, Delta As Double
RR = P * X / L
RL = P - RR
Delta = L / (N + 1)
```

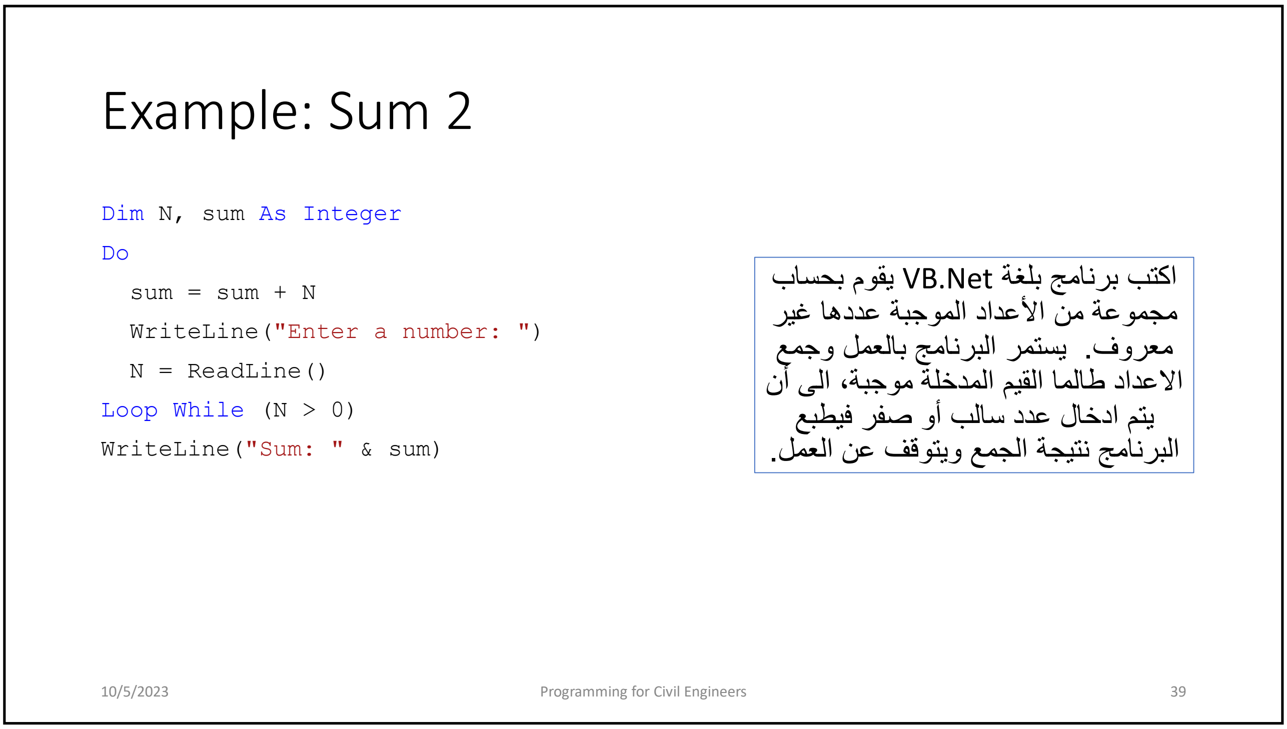
```
WriteLine("X" & Chr(9) & "M")
WriteLine("---" & Chr(9) & "---")
For D = 0 To L Step Delta

  If (D <= X) Then
    M = RL * D
  Else
    M = RR * (L - D)
  End If

  WriteLine(Round(D, 2) & Chr(9) & Round(M, 2))
Next
```



Example: Sum 1

```
Dim N, sum As Integer
WriteLine("Enter a number: ")
N = ReadLine()
sum = 0
Do While (N >= 0)
  sum = sum + N
  WriteLine("Enter a number: ")
  N = ReadLine()
Loop
WriteLine("Sum: " & sum)
```

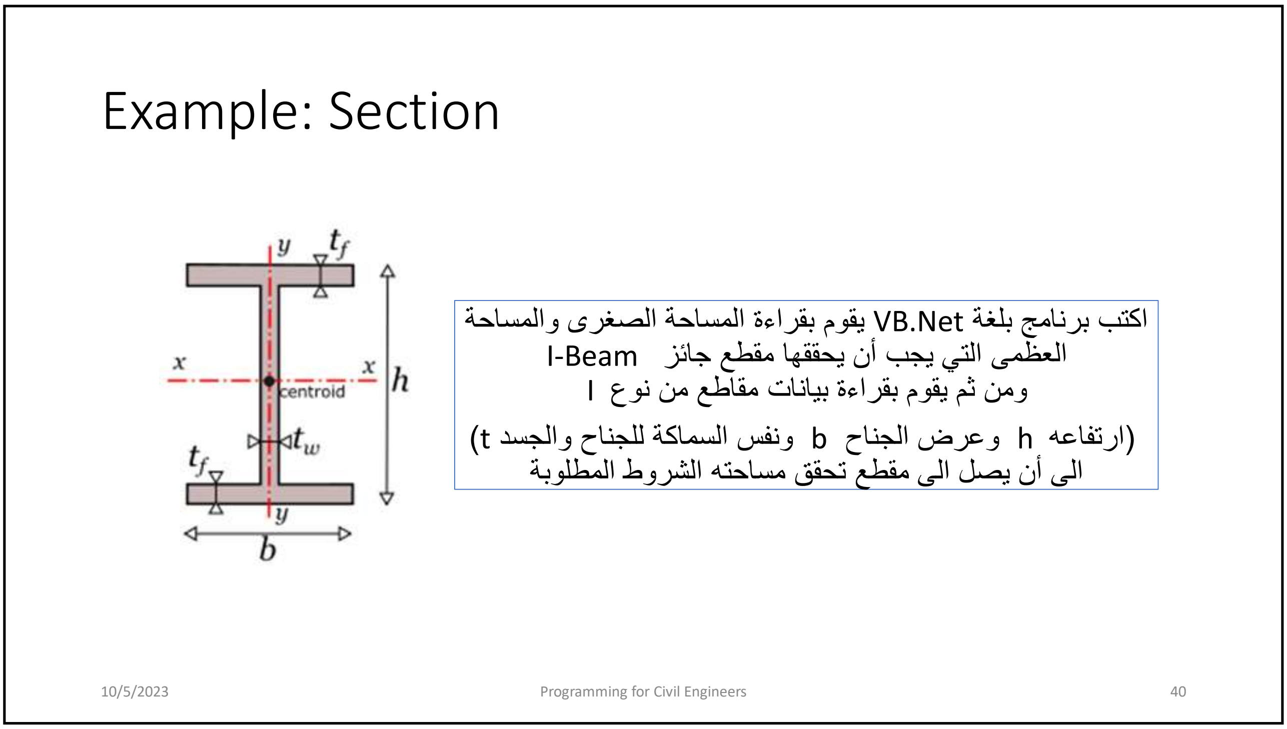
اكتب برنامج بلغة VB.Net يقوم بحساب مجموعة من الأعداد الموجبة عددها غير معروف. يستمر البرنامج بالعمل وجمع الاعداد طالما القيم المدخلة موجبة، الى أن يتم ادخال عدد سالب أو صفر فيطبع البرنامج نتيجة الجمع ويتوقف عن العمل.



Example: Sum 2

```
Dim N, sum As Integer
Do
  sum = sum + N
  WriteLine("Enter a number: ")
  N = ReadLine()
Loop While (N > 0)
WriteLine("Sum: " & sum)
```

اكتب برنامج بلغة VB.Net يقوم بحساب
مجموعة من الأعداد الموجبة عددها غير
معروف. يستمر البرنامج بالعمل وجمع
الاعداد طالما القيم المدخلة موجبة، الى أن
يتم ادخال عدد سالب أو صفر فيطبع
البرنامج نتيجة الجمع ويتوقف عن العمل.



Example: Section

اكتب برنامج بلغة VB.Net يقوم بقراءة المساحة الصغرى والمساحة
العظمى التي يجب أن يحققها مقطع جائز I-Beam
ومن ثم يقوم بقراءة بيانات مقاطع من نوع I
(ارتفاعه h وعرض الجناح b ونفس السماكة للجناح والجسد t)
الى أن يصل الى مقطع تحقق مساحته الشروط المطلوبة

```
Dim Amin, Amax, area, t, h, b As Double
WriteLine("Enter Minimum then Maximum Section Area in cm2: ")
Amin = ReadLine()
Amax = ReadLine()
Do
  WriteLine("Try a section: ")
  WriteLine("Enter Beam Section Width in mm: ")
  b = ReadLine()
  WriteLine("Enter Beam Section Height in mm: ")
  h = ReadLine()
  WriteLine("Enter Beam Section Thickness in mm: ")
  t = ReadLine()
  area = (2 * b * t + (h - 2 * t) * t) / 100
  WriteLine("Section Area= " & area & " cm2")
Loop Until (area >= Amin And area <= Amax)
WriteLine("Section is Acceptable")
```

Questions