People have always made models of the physical world, both to better understand it and to aid in project planning. Computers enable very complex systems to be modeled accurately. It is even possible to model so completely as to create a virtual reality.
A computer model represents characteristics of some physical phenomenon, and attempts to reproduce or approximate their behavior and/or outputs in response to inputs.

- Trivial example -- the Mac’s Audio CD Player, discussed as the “perfect GUI,” attempts to replicate a physical CD player to aid users in understanding its operation.
- Complex example -- Boeing’s 777 was designed on-line; certain parts such as the wing used sophisticated techniques computational fluid dynamics to create an “optimal” design.

The examples have dramatically different purposes.
Computer Modeling ...

- Physical world models are used in all areas of science and engineering, they are common in finance, production, marketing and most of business

- Modeling has the advantage that …
  - An artifact need not be created to be studied -- design
  - Complicated phenomena, too fast (explosive reactions), too tiny (molecular structure) or too distant (Jupiter) to observe, can be studied
  - Explanations for phenomena can be checked out before constructing an experiment or going on a field trip -- exploration

- Modeling’s main disadvantage: It’s only as good as the model
Constructing A Computer Model

- In a model all phenomena are represented as data (mostly numbers) and all changes are realized by computations on the numbers
  - To model the atmospheric pressure of an aircraft’s surface as it returns from space flight, the geometry of the plane must be represented in a computer: polygons
  - The Navier-Stokes equations for air flow must be applied
Simulation: Making A Model Behave

- An essential tool of computer modeling is simulation.
- A bank can be modeled using simulated tellers and depositors and mathematics called “queuing theory.”
- Select a random arrival rate (frequency of new customers, and service rate -- teller speed)

 Servers

 Arrivals

 Queue
A key issue in a simulation is the representation of the physical phenomena. It is not necessary to give form to all aspects of the phenomena, only those features necessary for the computation.

To model the orbits of the planets of the solar system, it is only necessary to know their positions in space \((x,y,z)\) and their mass.
Raindrops

- Consider modeling raindrops on pavement.
- The model will be a square region of pavement 2000 x 2000 units and randomly sized raindrops modeled as circles with diameter chosen between 100 - 200.

```vbnet
Private Sub Form_Click()
    Dim xCoord As Integer, yCoord As Integer
    Dim rad As Integer
    clickCount = clickCount + 1
    If clickCount = 1 Then
        Line (1000, 1000)-(3000, 3000), , B
    Else
        FillColor = RGB(0, 0, 255)
        FillStyle = 0
        xCoord = Int(Rnd(1) * 2000) + 1000
        yCoord = Int(Rnd(1) * 2000) + 1000
        rad = Int(Rnd(1) * 100)
        Circle (xCoord, yCoord), rad + 100, RGB(0, 0, 255)
    End If
End Sub
```

Clicking to fill in raindrops allows us to follow action.
Automated Rain

- Use a loop to make rain, perhaps generating a batch with each click.

How many raindrops does it take to cover the region of the pavement?

There is no one answer, but rather a range.

```
index = 0
Do While index < 100
   index = index + 1
   xCoord = Int(Rnd(1) * 2000) + 1000
   yCoord = Int(Rnd(1) * 2000) + 1000
   rad = Int(Rnd(1) * 100)
   Circle (xCoord, yCoord), rad + 100, RGB(0, 0, 255)
```

100 drops
200 drops
300 drops
How To Solve It

- Generate raindrops until all of the pavement is covered while counting them … expect a loop with a termination test based on “all wet”

- How to determine if pavement is wet?
  - The program cannot “look” at the form and “see” if it is “wet”
  - How is a position on the pavement determined to be wet?
    - Being under a droplet
  - When is the position “under” the droplet?
    - When it is within “rad” of the circle center
  - How is “within” rad determined?
    - Distance between pavement position (x, y) and center (xCoord, yCoord) is
      \[ \sqrt{(xCoord-x)^2 + (yCoord-y)^2} \]
What Positions To Inspect

- Cannot look at every point since there are infinitely many points on the pavement (in nature)
- Make a “discrete approximation” by checking points every few units, say every four, in a regular grid pattern
- If every grid point is covered by some drop, then the pavement will be defined to be wet
- This is only an approximation
How To Check Grid Points

- Recall that raindrops are generated until “all wet”
- To get started, we could generate until the first grid point gets wet
- Logical steps
  - Generate a raindrop and store it in the array
  - Check if that raindrop covers up the first grid point (1000,1000)
    - If yes, keep looking at successive points
    - If no, loop around and create another point
  - Check until a dry point is found or all points have been check
  - Must keep a list of the raindrop positions
Variables

- Choose the size of the arrays to be “sufficiently large”

```
Private Sub Form_Load()
    Randomize
    clickCount = 0
End Sub

Private Sub Form_Click()
    Dim xCoord(2999) As Integer, yCoord(2999) As Integer
    Dim rad(2999) As Integer
    Dim gPointx As Integer, gPointy As Integer
    Dim index As Integer, nMax As Integer
    gPointx = 1000
    gPointy = 1000
    nMax = 0
    index = 0
```

Variables gPointx and gPointy are the grid point being examined; index is the droplet being examined; nMax is the total number of droplets.
Logic

- Sweeping through the grid points requires that they be enumerated ...

1000 < gPointy < 3000
1000 < gPointx < 3000

Test if (gPointx, gPointy) wet
Yes: gPointx = gPointx + 4
No: Test if there are unexamined droplets
   Yes: index = index + 1
   No: nMax = nMax + 1

Generate droplet

gPointx = 0
index = 0

1000 < gPointy < 3000

Examine all grid points
Wet test
Advance in row
Still droplets to go
Test it
Create another

nMax = nMax + 1
Generate droplet

Restart on column
Restart on droplets
Move to next row
Program Text

Do While gPointx < 3000
    If rad(index) > ((xCoord(index) - gPointx) ^ 2 + (yCoord(index) - gPointy) ^ 2) ^ 0.5 Then
        gPointx = gPointx + 4
    Else
        If index < nMax Then
            index = index + 1
            FillColor = RGB(255, 0, 0)
        Else
            nMax = nMax + 1
            index = index + 1
            xCoord(index) = Int(Rnd(1) * 2000) + 1000
            yCoord(index) = Int(Rnd(1) * 2000) + 1000
            rad(index) = Int(Rnd(1) * 100) + 100
            Circle (xCoord(index), yCoord(index)), 20, RGB(0, 255, 255)
        End If
    End If
End If
Loop

The inner loop moving across a row of the grid
Outer Loop

The droplet array is initialized to have its first droplet before entering the loop.
Executing The Program

- As feedback, trace the droplets generated trying to get the first “hit” (in aqua) and then the droplets added throughout the remainder of the computation (red)
A computer model is only as good as the mathematics and programming on which it is founded. All computer models ignore features of the physical system and all make simplifying assumptions. A computer model’s predictive ability is directly related to the features ignored and the assumptions made … so, do not automatically accept a computer model any more than you would automatically accept a legal contract … Check the fine print!