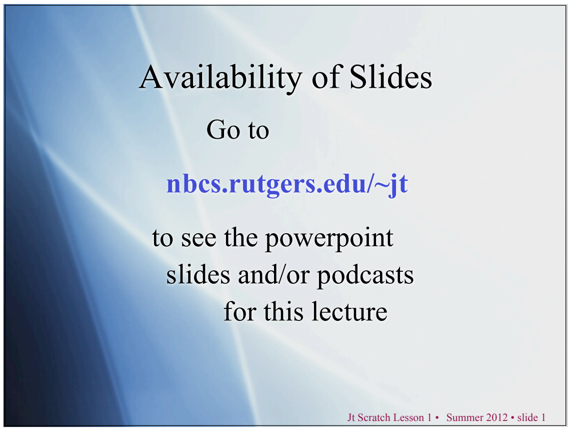


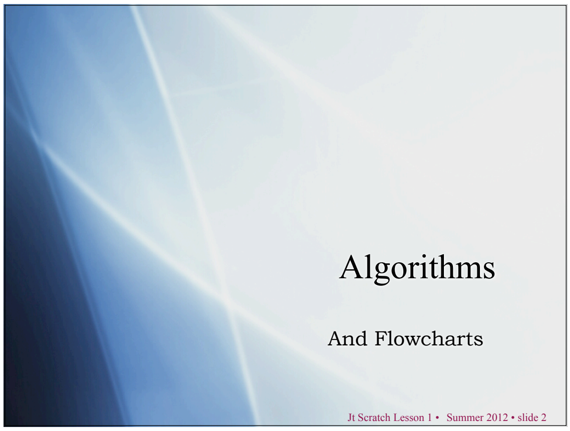
Availability of Slides

Go to

nbcs.rutgers.edu/~jt

to see the powerpoint
slides and/or podcasts
for this lecture

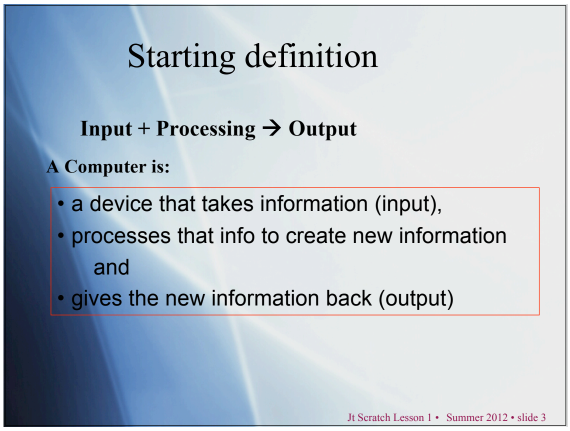
Jt Scratch Lesson 1 • Summer 2012 • slide 1



Algorithms

And Flowcharts

Jt Scratch Lesson 1 • Summer 2012 • slide 2



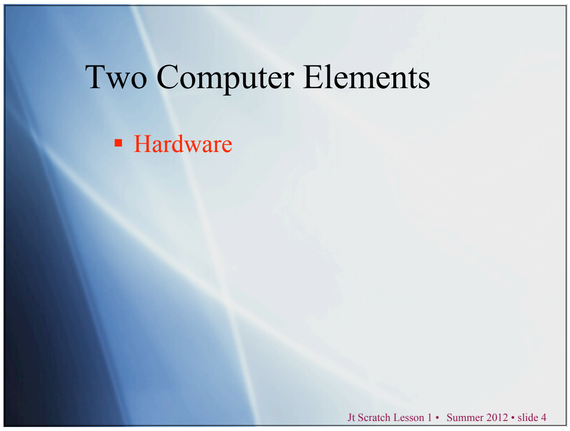
Starting definition

Input + Processing → Output

A Computer is:

- a device that takes information (input),
- processes that info to create new information and
- gives the new information back (output)

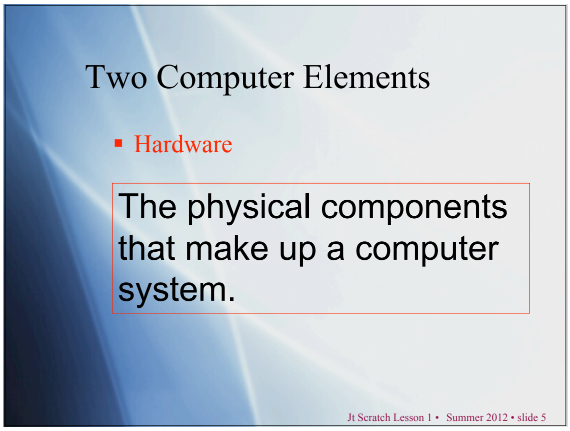
Jt Scratch Lesson 1 • Summer 2012 • slide 3



Two Computer Elements

- Hardware

Jt Scratch Lesson 1 • Summer 2012 • slide 4

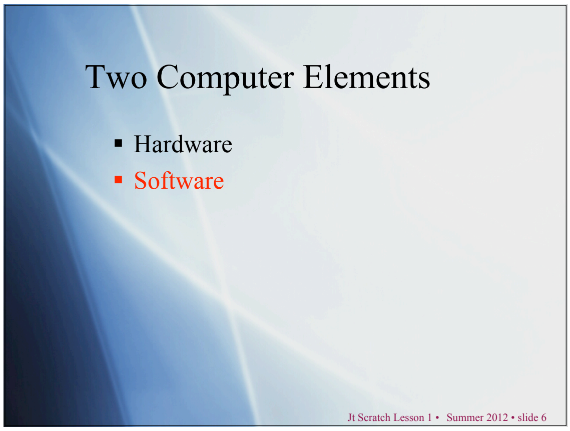


Two Computer Elements

- Hardware

The physical components that make up a computer system.

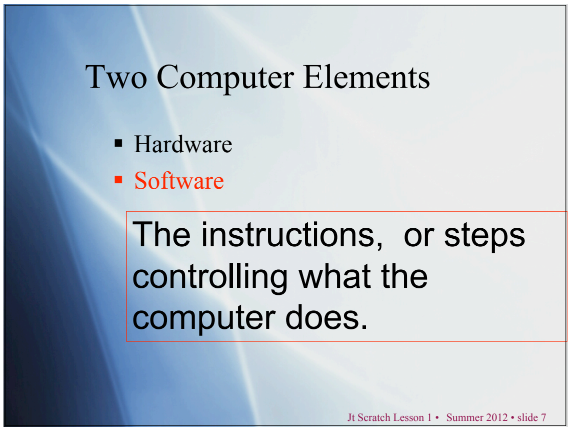
Jt Scratch Lesson 1 • Summer 2012 • slide 5



Two Computer Elements

- Hardware
- Software

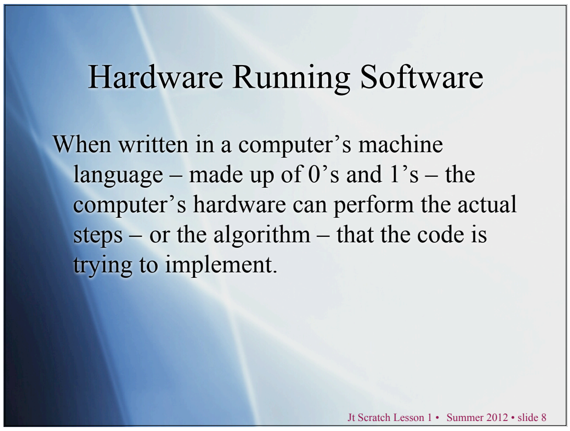
Jt Scratch Lesson 1 • Summer 2012 • slide 6



Two Computer Elements

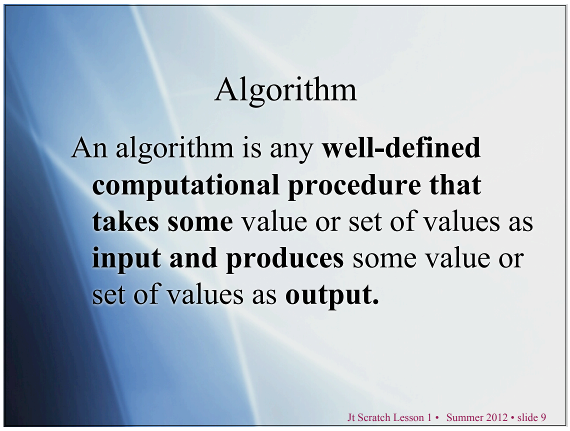
- Hardware
- Software

The instructions, or steps controlling what the computer does.



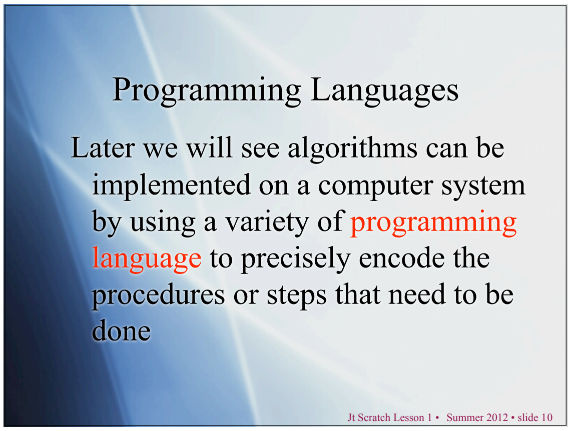
Hardware Running Software

When written in a computer's machine language – made up of 0's and 1's – the computer's hardware can perform the actual steps – or the algorithm – that the code is trying to implement.



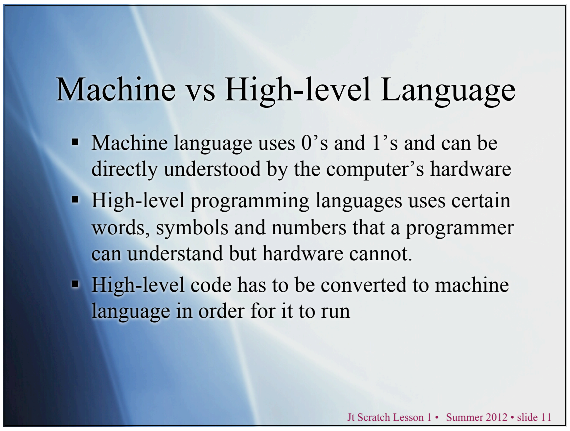
Algorithm

An algorithm is any **well-defined computational procedure that takes some** value or set of values as **input and produces** some value or set of values as **output.**



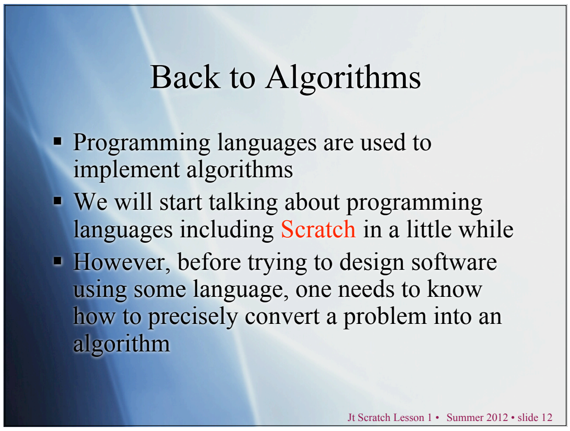
Programming Languages

Later we will see algorithms can be implemented on a computer system by using a variety of programming language to precisely encode the procedures or steps that need to be done



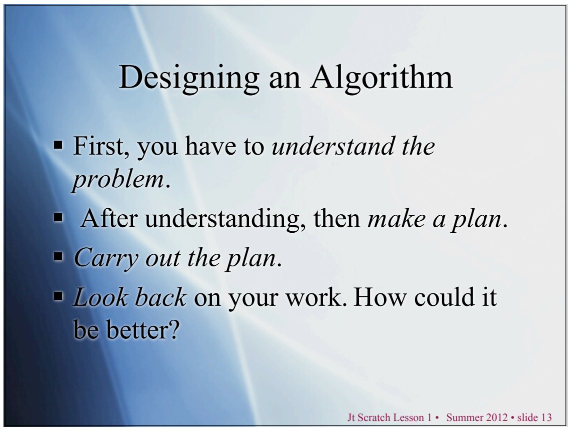
Machine vs High-level Language

- Machine language uses 0's and 1's and can be directly understood by the computer's hardware
- High-level programming languages uses certain words, symbols and numbers that a programmer can understand but hardware cannot.
- High-level code has to be converted to machine language in order for it to run



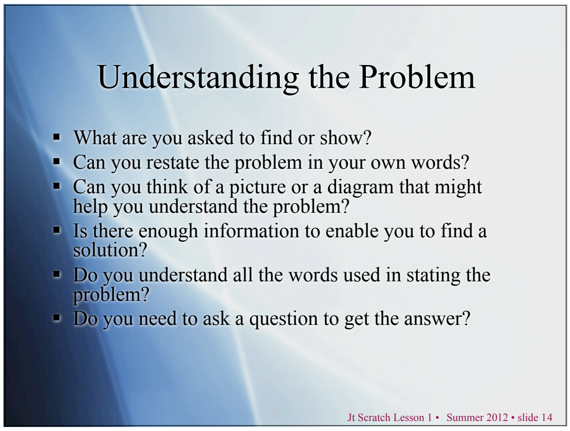
Back to Algorithms

- Programming languages are used to implement algorithms
- We will start talking about programming languages including Scratch in a little while
- However, before trying to design software using some language, one needs to know how to precisely convert a problem into an algorithm



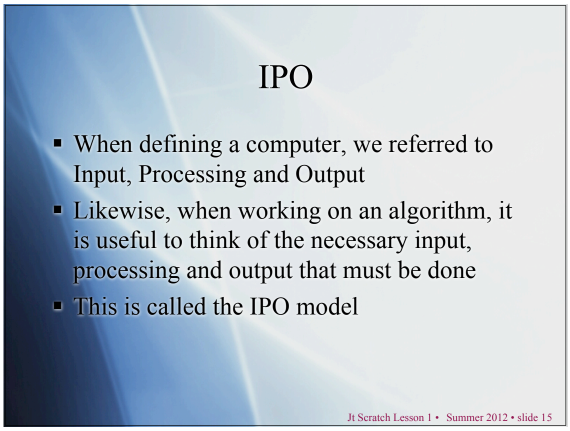
Designing an Algorithm

- First, you have to *understand the problem*.
- After understanding, then *make a plan*.
- *Carry out the plan.*
- *Look back* on your work. How could it be better?



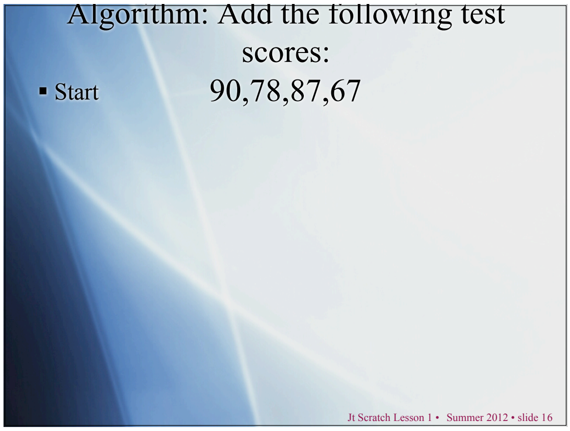
Understanding the Problem

- What are you asked to find or show?
- Can you restate the problem in your own words?
- Can you think of a picture or a diagram that might help you understand the problem?
- Is there enough information to enable you to find a solution?
- Do you understand all the words used in stating the problem?
- Do you need to ask a question to get the answer?



IPO

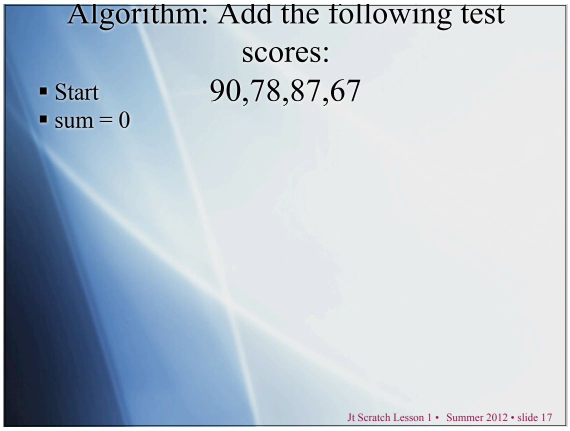
- When defining a computer, we referred to Input, Processing and Output
- Likewise, when working on an algorithm, it is useful to think of the necessary input, processing and output that must be done
- This is called the IPO model



Algorithm: Add the following test scores:

90,78,87,67

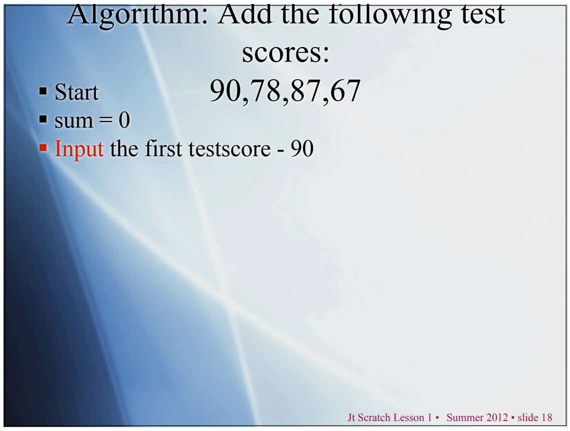
- Start



Algorithm: Add the following test scores:

90,78,87,67

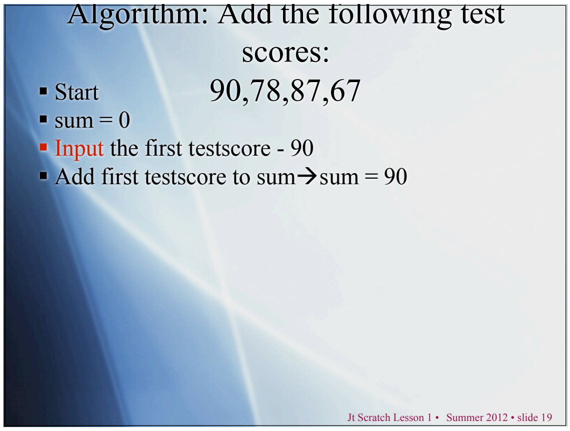
- Start
- sum = 0



Algorithm: Add the following test scores:

90,78,87,67

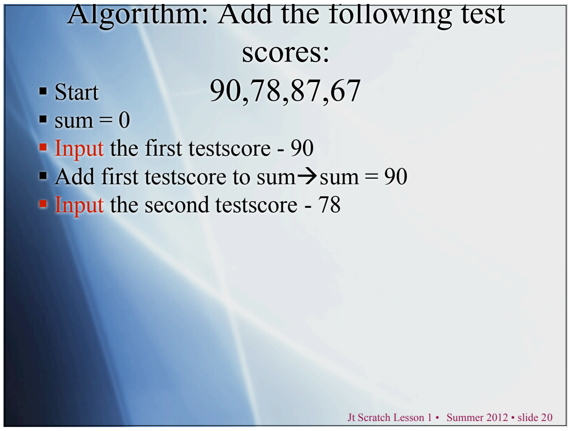
- Start
- sum = 0
- Input the first testscore - 90



Algorithm: Add the following test scores:

90,78,87,67

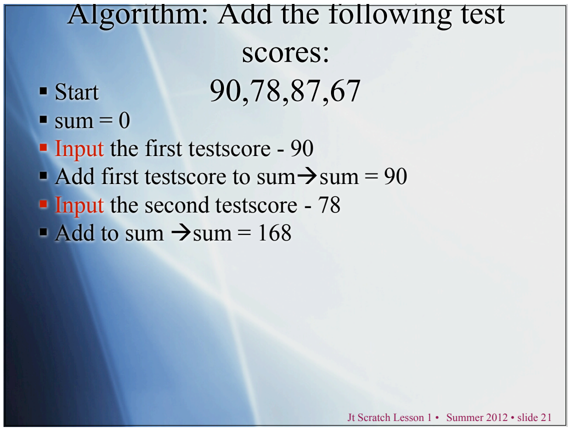
- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90



Algorithm: Add the following test scores:

90,78,87,67

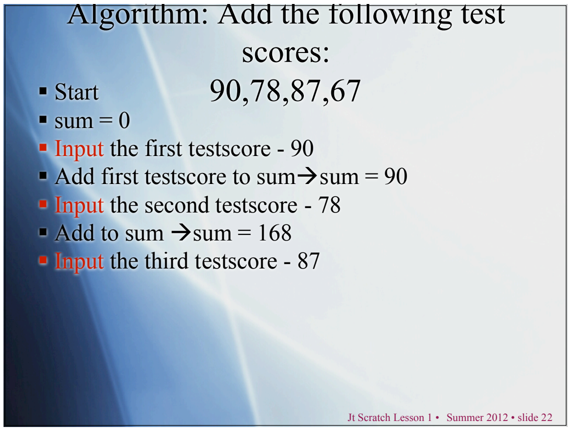
- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78



Algorithm: Add the following test scores:

90,78,87,67

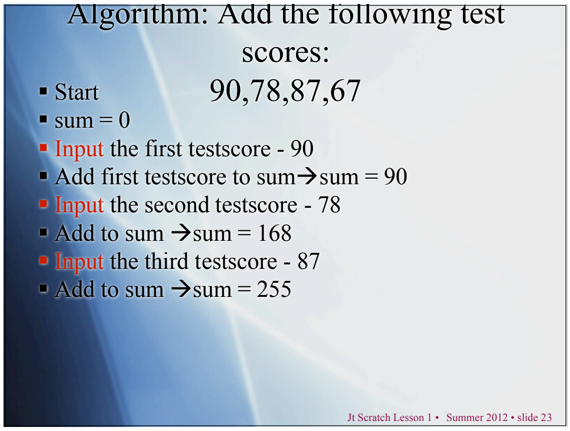
- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168



Algorithm: Add the following test scores:

90,78,87,67

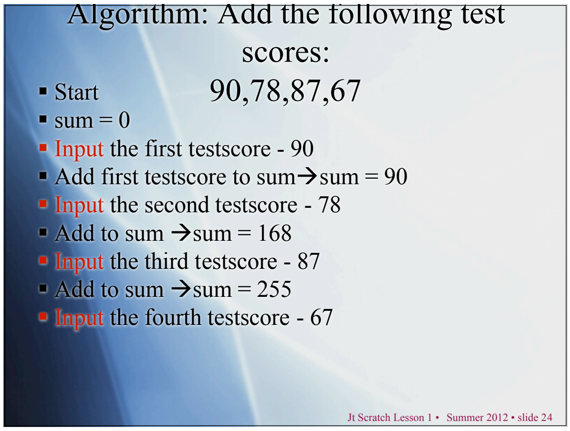
- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168
- Input the third testscore - 87



Algorithm: Add the following test scores:

90,78,87,67

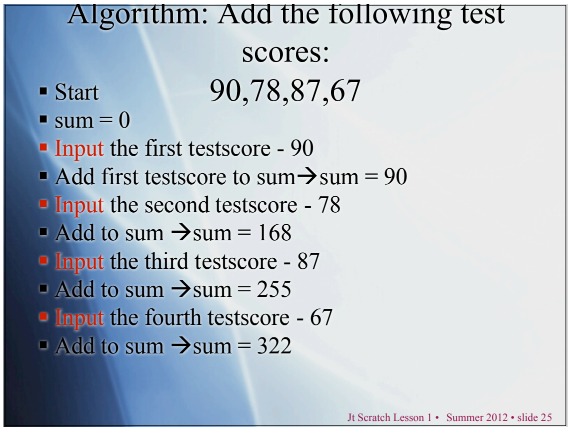
- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168
- Input the third testscore - 87
- Add to sum →sum = 255



Algorithm: Add the following test scores:

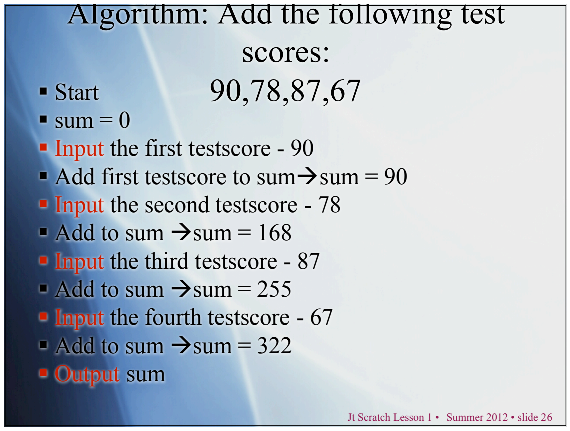
90,78,87,67

- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168
- Input the third testscore - 87
- Add to sum →sum = 255
- Input the fourth testscore - 67



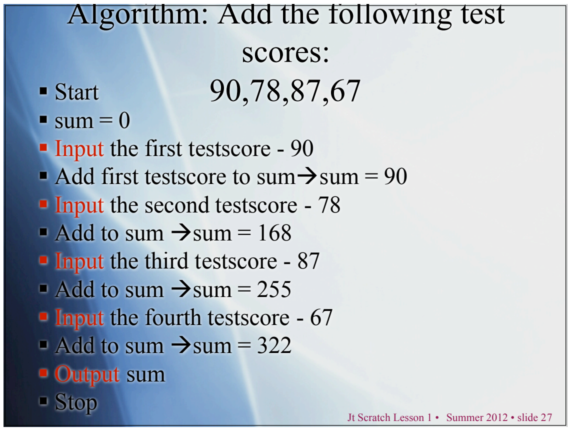
Algorithm: Add the following test scores: 90,78,87,67

- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168
- Input the third testscore - 87
- Add to sum →sum = 255
- Input the fourth testscore - 67
- Add to sum →sum = 322



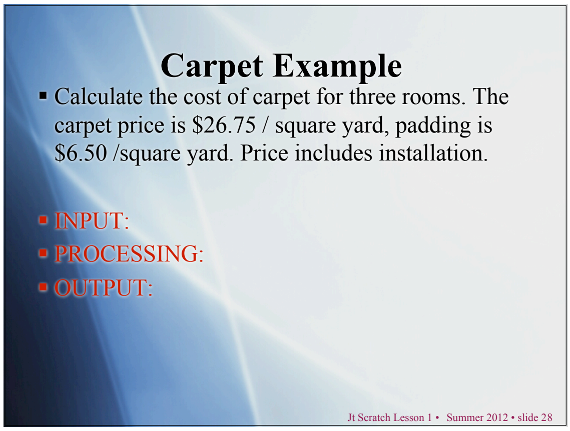
Algorithm: Add the following test scores: 90,78,87,67

- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168
- Input the third testscore - 87
- Add to sum →sum = 255
- Input the fourth testscore - 67
- Add to sum →sum = 322
- Output sum



Algorithm: Add the following test scores: 90,78,87,67

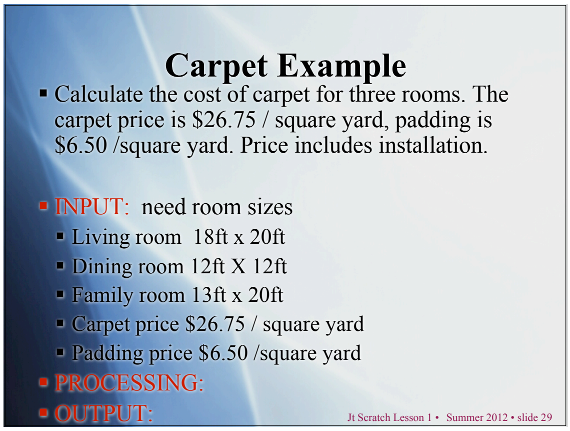
- Start
- sum = 0
- Input the first testscore - 90
- Add first testscore to sum→sum = 90
- Input the second testscore - 78
- Add to sum →sum = 168
- Input the third testscore - 87
- Add to sum →sum = 255
- Input the fourth testscore - 67
- Add to sum →sum = 322
- Output sum
- Stop



Carpet Example

- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.

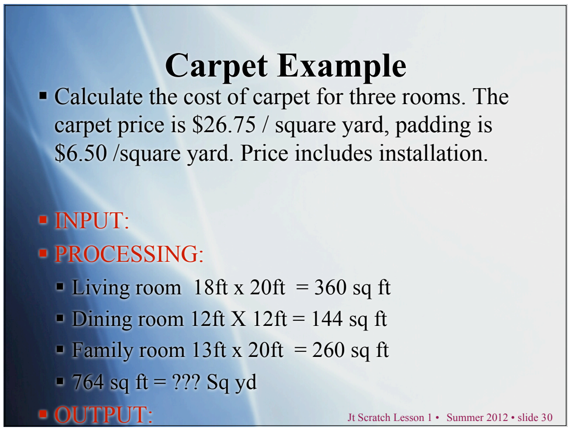
- INPUT:
- PROCESSING:
- OUTPUT:



Carpet Example

- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.

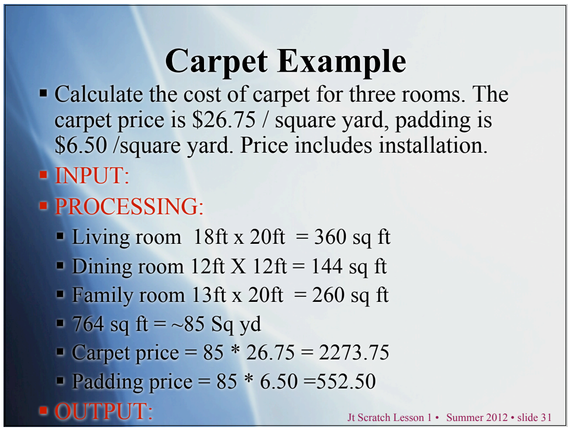
- INPUT: need room sizes
  - Living room 18ft x 20ft
  - Dining room 12ft X 12ft
  - Family room 13ft x 20ft
  - Carpet price $26.75 / square yard
  - Padding price $6.50 /square yard
- PROCESSING:
- OUTPUT:



Carpet Example

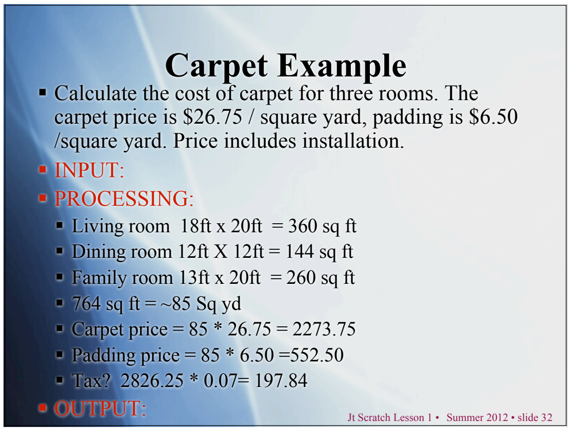
- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.

- INPUT:
- PROCESSING:
  - Living room 18ft x 20ft = 360 sq ft
  - Dining room 12ft X 12ft = 144 sq ft
  - Family room 13ft x 20ft = 260 sq ft
  - 764 sq ft = ??? Sq yd
- OUTPUT:



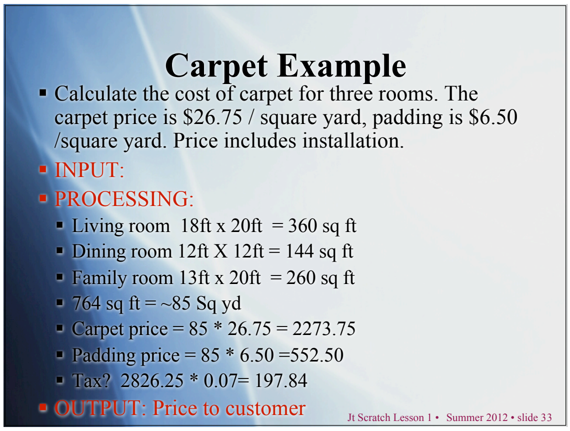
Carpet Example

- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.
- INPUT:
- PROCESSING:
  - Living room 18ft x 20ft = 360 sq ft
  - Dining room 12ft X 12ft = 144 sq ft
  - Family room 13ft x 20ft = 260 sq ft
  - 764 sq ft = ~85 Sq yd
  - Carpet price = 85 * 26.75 = 2273.75
  - Padding price = 85 * 6.50 =552.50
- OUTPUT:



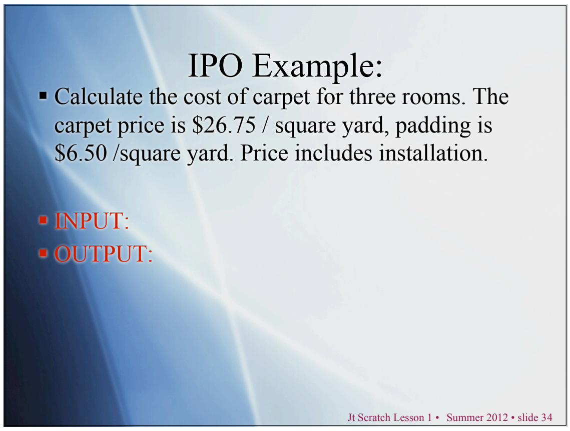
Carpet Example

- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.
- INPUT:
- PROCESSING:
  - Living room 18ft x 20ft = 360 sq ft
  - Dining room 12ft X 12ft = 144 sq ft
  - Family room 13ft x 20ft = 260 sq ft
  - 764 sq ft = ~85 Sq yd
  - Carpet price = 85 * 26.75 = 2273.75
  - Padding price = 85 * 6.50 =552.50
  - Tax? 2826.25 * 0.07= 197.84
- OUTPUT:



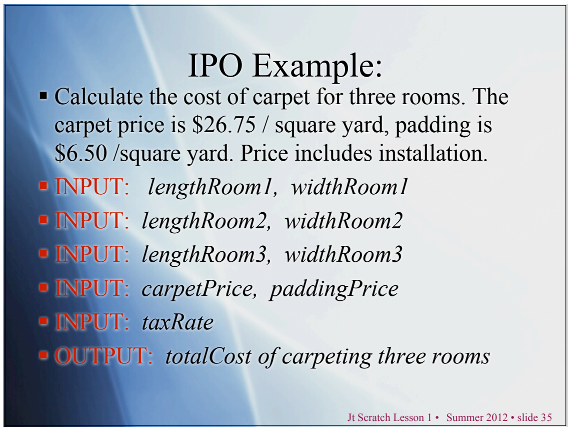
Carpet Example

- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.
- INPUT:
- PROCESSING:
  - Living room 18ft x 20ft = 360 sq ft
  - Dining room 12ft X 12ft = 144 sq ft
  - Family room 13ft x 20ft = 260 sq ft
  - 764 sq ft = ~85 Sq yd
  - Carpet price = 85 * 26.75 = 2273.75
  - Padding price = 85 * 6.50 =552.50
  - Tax? 2826.25 * 0.07= 197.84
- OUTPUT: Price to customer



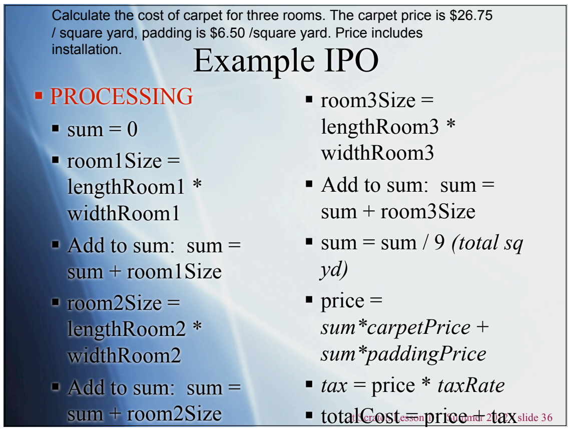
IPO Example:

- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.
- INPUT:
- OUTPUT:



IPO Example:

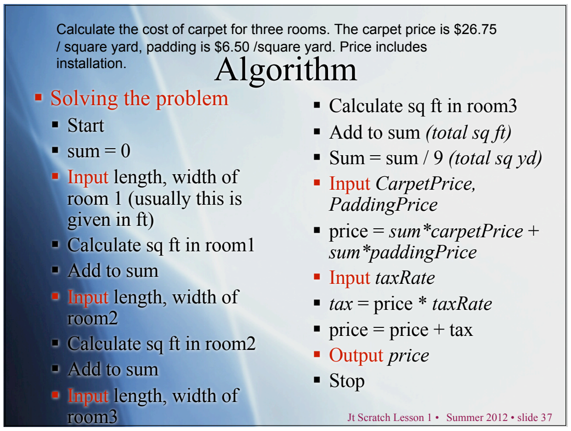
- Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.
- INPUT: *lengthRoom1, widthRoom1*
- INPUT: *lengthRoom2, widthRoom2*
- INPUT: *lengthRoom3, widthRoom3*
- INPUT: *carpetPrice, paddingPrice*
- INPUT: *taxRate*
- OUTPUT: *totalCost of carpeting three rooms*



Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.

Example IPO

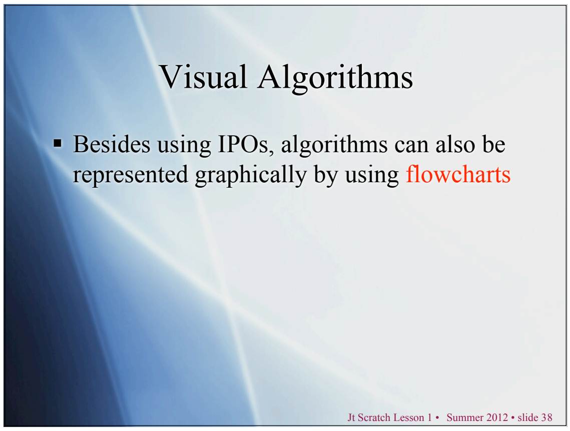
- PROCESSING
  - sum = 0
  - room1Size = lengthRoom1 * widthRoom1
  - Add to sum: sum = sum + room1Size
  - room2Size = lengthRoom2 * widthRoom2
  - Add to sum: sum = sum + room2Size
  - room3Size = lengthRoom3 * widthRoom3
  - Add to sum: sum = sum + room3Size
  - sum = sum / 9 *(total sq yd)*
  - price = *sum*carpetPrice* + *sum*paddingPrice*
  - *tax* = price * *taxRate*
  - totalCost = price + tax



Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.

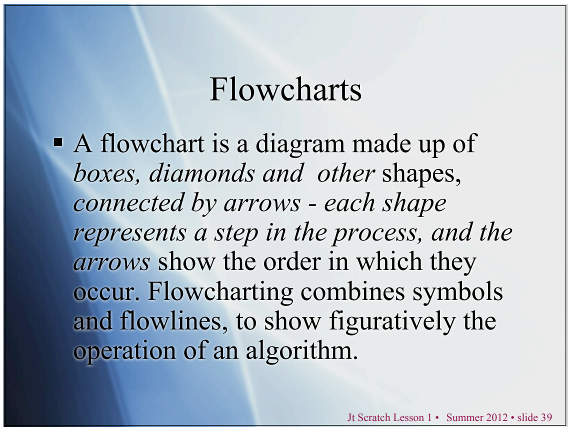
Algorithm

- Solving the problem
  - Start
  - sum = 0
  - Input length, width of room 1 (usually this is given in ft)
  - Calculate sq ft in room1
  - Add to sum
  - Input length, width of room2
  - Calculate sq ft in room2
  - Add to sum
  - Input length, width of room3
  - Calculate sq ft in room3
  - Add to sum *(total sq ft)*
  - Sum = sum / 9 *(total sq yd)*
  - Input *CarpetPrice, PaddingPrice*
  - price = *sum*carpetPrice* + *sum*paddingPrice*
  - Input *taxRate*
  - *tax* = price * *taxRate*
  - price = price + tax
  - Output *price*
  - Stop



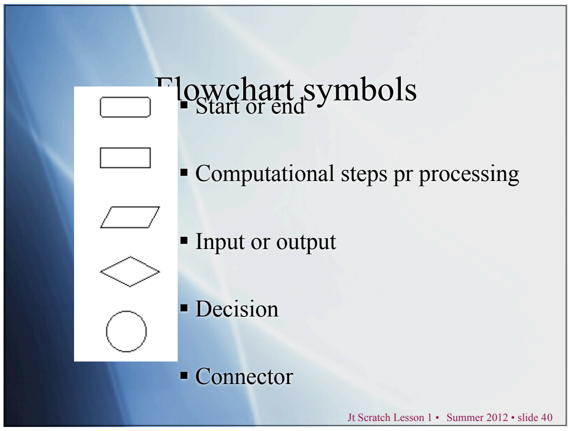
Visual Algorithms

- Besides using IPOs, algorithms can also be represented graphically by using flowcharts



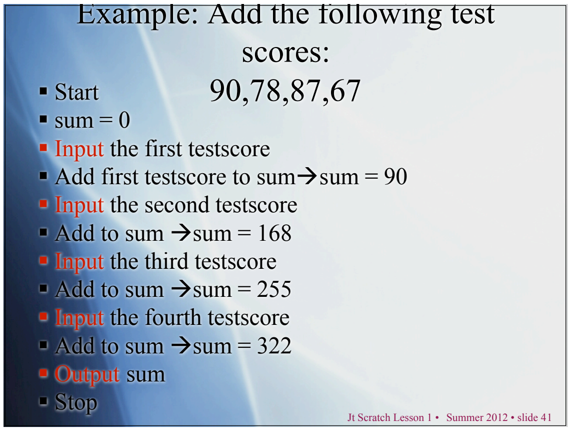
Flowcharts

- A flowchart is a diagram made up of *boxes, diamonds and other* shapes, *connected by arrows - each shape represents a step in the process, and the arrows* show the order in which they occur. Flowcharting combines symbols and flowlines, to show figuratively the operation of an algorithm.



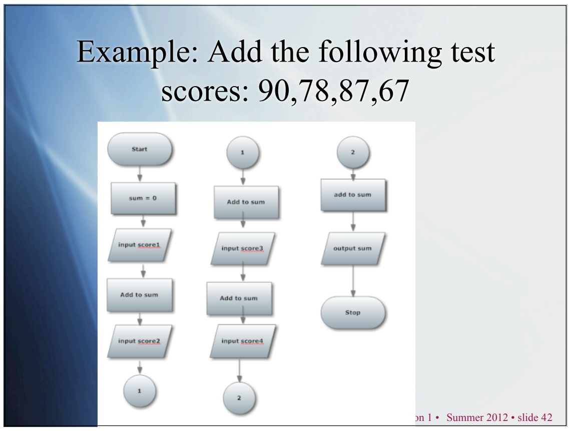
Flowchart symbols

- Start or end
- Computational steps pr processing
- Input or output
- Decision
- Connector



Example: Add the following test scores: 90,78,87,67

- Start
- sum = 0
- Input the first testscore
- Add first testscore to sum→sum = 90
- Input the second testscore
- Add to sum →sum = 168
- Input the third testscore
- Add to sum →sum = 255
- Input the fourth testscore
- Add to sum →sum = 322
- Output sum
- Stop



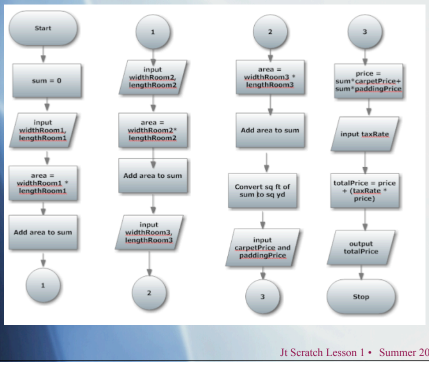
Example: Add the following test scores: 90,78,87,67

Calculate the cost of carpet for three rooms. The carpet price is $26.75 / square yard, padding is $6.50 /square yard. Price includes installation.

Example

- Solving the problem
  - Start
  - sum = 0
  - Input length, width of room 1 (usually this is given in ft)
  - Calculate sq ft in room1
  - Add to sum
  - Input length, width of room2
  - Calculate sq ft in room2
  - Add to sum
  - Input length, width of room3
  - Calculate sq ft in room3
  - Add to sum *(total sq ft)*
  - Sum = sum / 9 *(total sq yd)*
  - Input *CarpetPrice, PaddingPrice*
  - price = *sum*carpetPrice + sum*paddingPrice*
  - Input *taxRate*
  - *tax* = price * *taxRate*
  - totalPrice = price + tax
  - Output *price*
  - Stop

Carpet buying



Start
sum = 0
input widthRoom1, lengthRoom1
area = widthRoom1 * lengthRoom1
Add area to sum
1

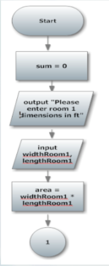
1
input widthRoom2, lengthRoom2
area = widthRoom2* lengthRoom2
Add area to sum
input widthRoom3, lengthRoom3
2

2
area = widthRoom3 * lengthRoom3
Add area to sum
Convert sq ft of sum to sq yd
input carpetPrice and paddingPrice
3

3
price = sum*carpetPrice+ sum*paddingPrice
input taxRate
totalPrice = price + (taxRate * price)
output totalPrice
Stop

More *output*

You will really ASK the user to enter values



Start
sum = 0
output "Please enter room 1 dimensions in ft"
input widthRoom1, lengthRoom1
area = widthRoom1 * lengthRoom1
1

Programming Languages

- IPOs and Flowcharts are useful to think through an algorithm
- But to get them to work on a computer, you need to encode them using a programming language
- Which ultimately is converted – or compiled – into the computer's machine langaueg

Programming Languages

- Hundreds of them:
  - Early ones: BASIC, Fortran, Cobol
  - Others: PHP, Java, C++
  - What we are using: Scratch

Availability of Slides

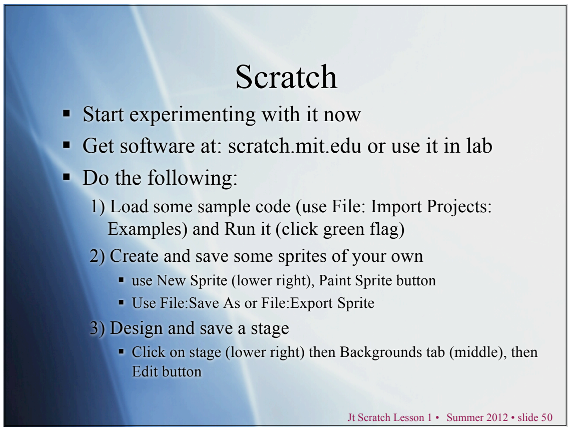
Go to

nbcs.rutgers.edu/~jt

to see the powerpoint
slides and/or podcasts
for this lecture

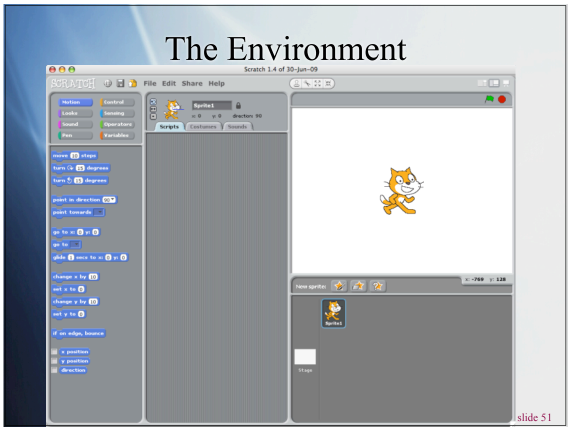
Scratch Programming

Lesson 1: Introduction

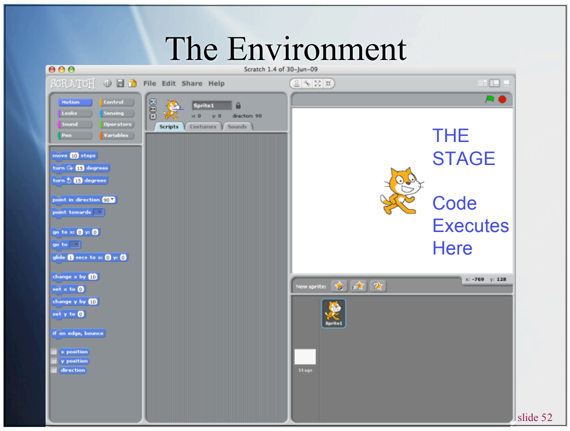


Scratch

- Start experimenting with it now
- Get software at: scratch.mit.edu or use it in lab
- Do the following:
  1) Load some sample code (use File: Import Projects: Examples) and Run it (click green flag)
  2) Create and save some sprites of your own
     - use New Sprite (lower right), Paint Sprite button
     - Use File:Save As or File:Export Sprite
  3) Design and save a stage
     - Click on stage (lower right) then Backgrounds tab (middle), then Edit button



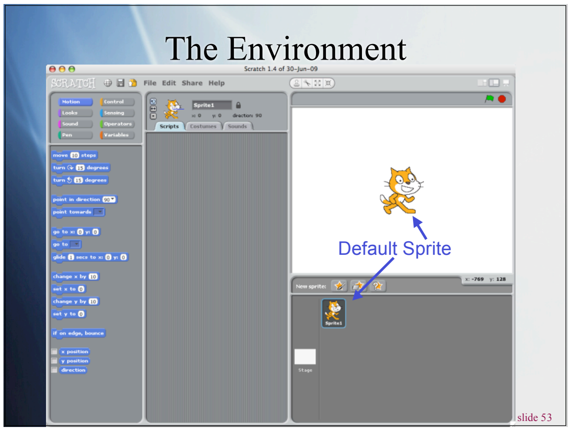
The Environment



The Environment

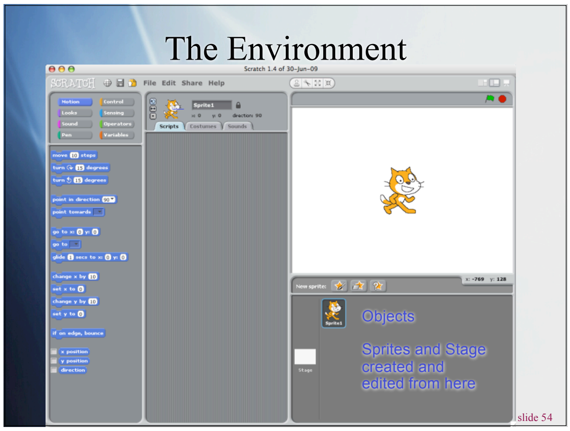
THE STAGE

Code Executes Here



The Environment

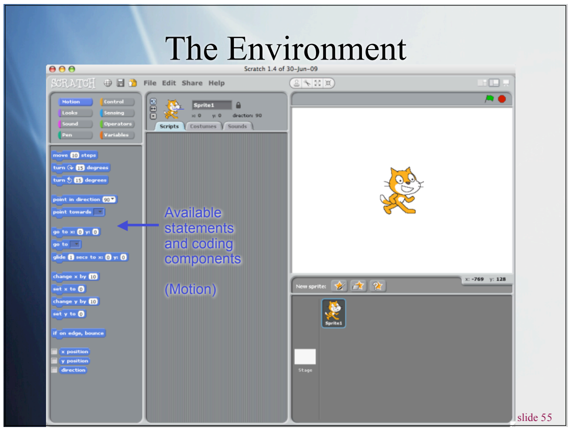
Default Sprite



The Environment

Objects

Sprites and Stage created and edited from here

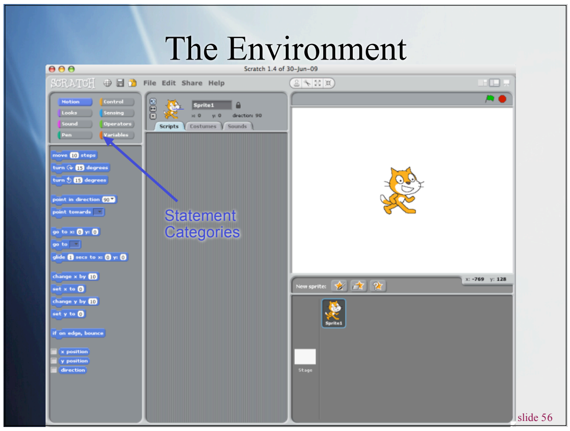


The Environment

Available statements and coding components

(Motion)

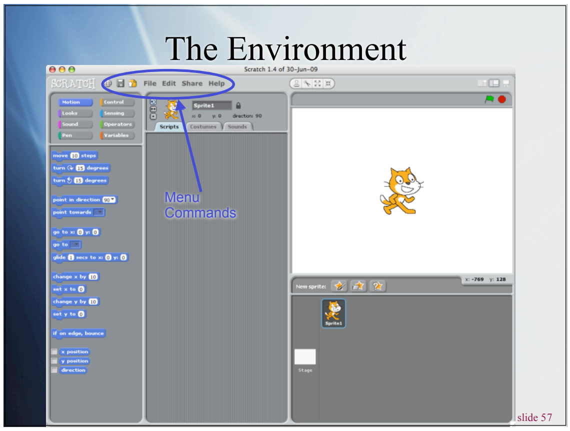
slide 55



The Environment

Statement Categories

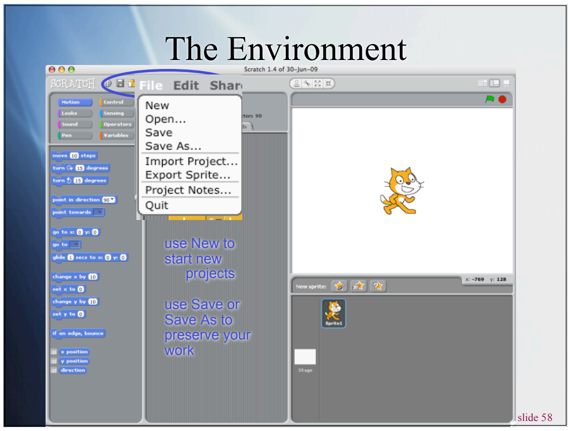
slide 56



The Environment

Menu Commands

slide 57

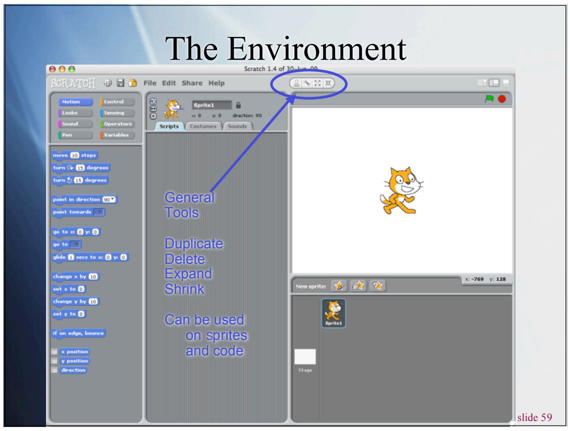


The Environment

use New to start new projects

use Save or Save As to preserve your work

slide 58



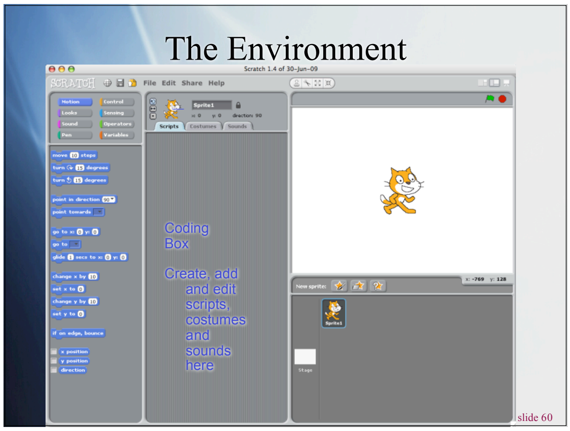
The Environment

General Tools

Duplicate
Delete
Expand
Shrink

Can be used on sprites and code

slide 59

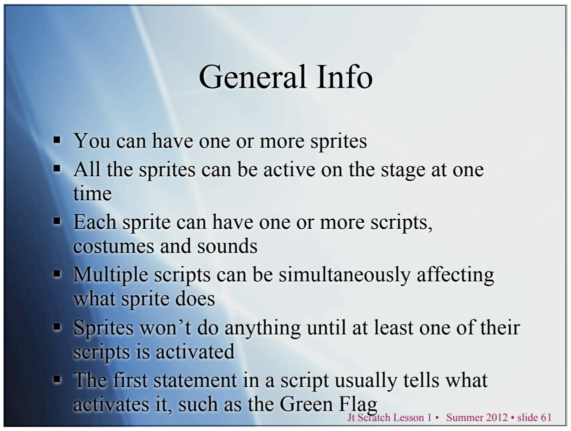


The Environment

Coding Box

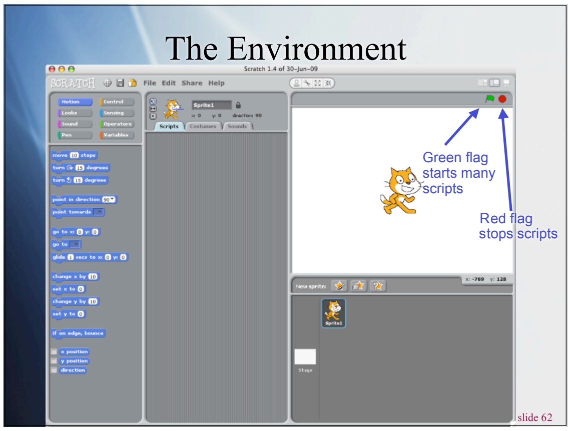
Create, add and edit scripts, costumes and sounds here

slide 60



General Info

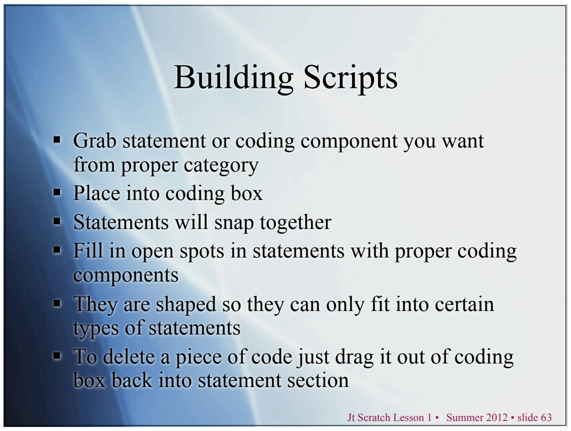
- You can have one or more sprites
- All the sprites can be active on the stage at one time
- Each sprite can have one or more scripts, costumes and sounds
- Multiple scripts can be simultaneously affecting what sprite does
- Sprites won't do anything until at least one of their scripts is activated
- The first statement in a script usually tells what activates it, such as the Green Flag



The Environment

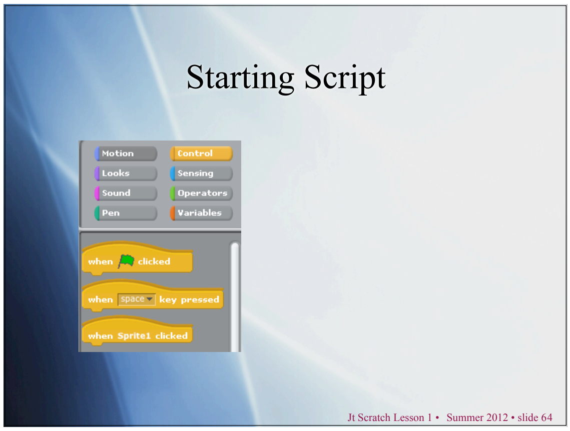
Green flag starts many scripts

Red flag stops scripts

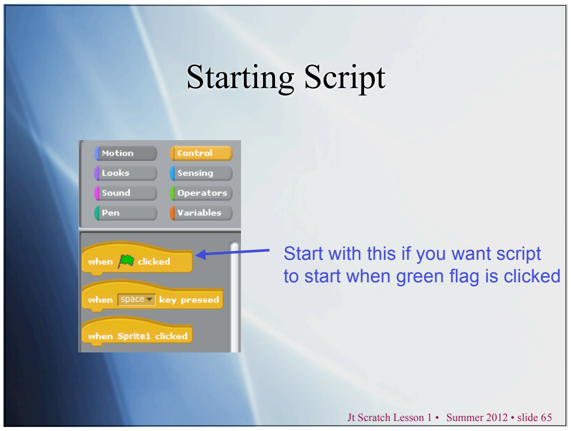


Building Scripts

- Grab statement or coding component you want from proper category
- Place into coding box
- Statements will snap together
- Fill in open spots in statements with proper coding components
- They are shaped so they can only fit into certain types of statements
- To delete a piece of code just drag it out of coding box back into statement section

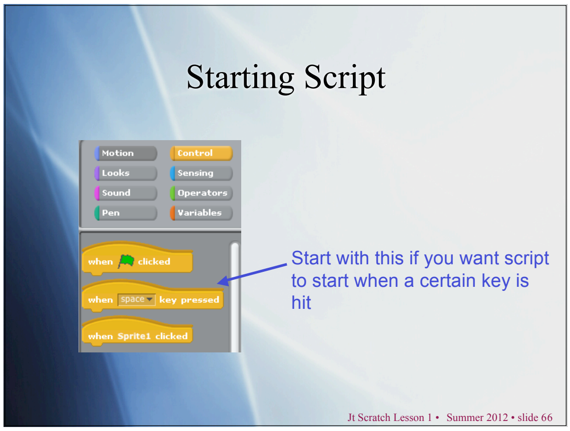


Starting Script



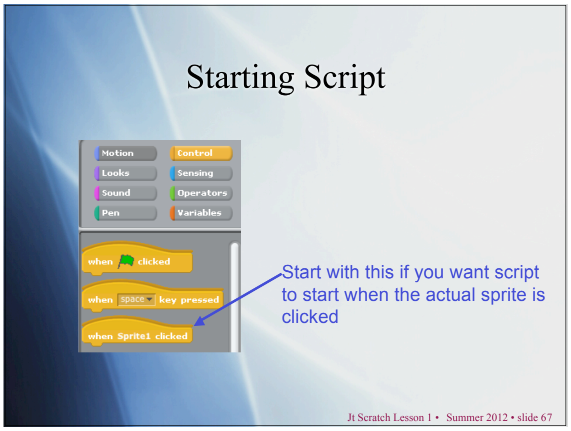
Starting Script

Start with this if you want script to start when green flag is clicked



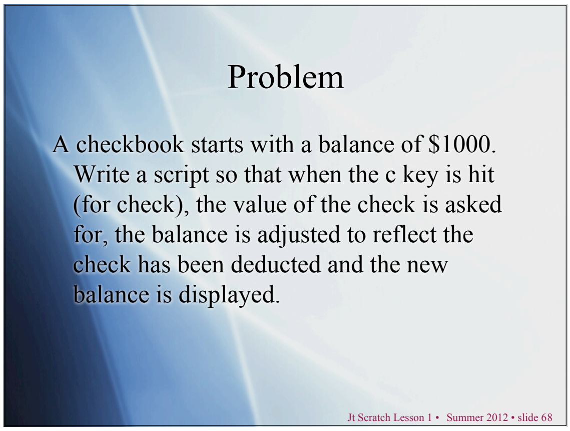
Starting Script

Start with this if you want script to start when a certain key is hit



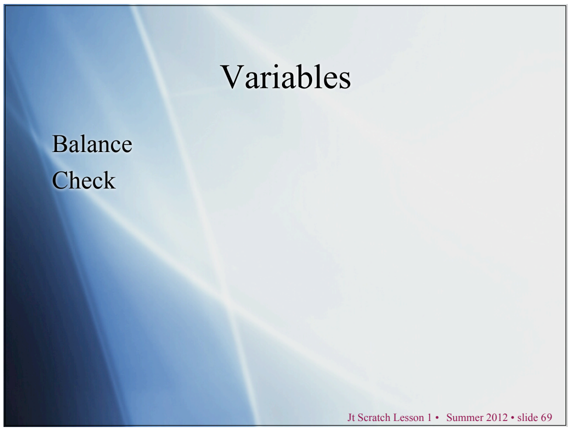
Starting Script

Start with this if you want script to start when the actual sprite is clicked



Problem

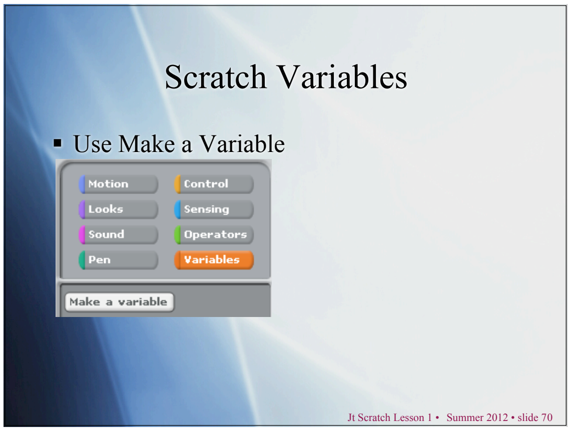
A checkbook starts with a balance of $1000. Write a script so that when the c key is hit (for check), the value of the check is asked for, the balance is adjusted to reflect the check has been deducted and the new balance is displayed.



Variables

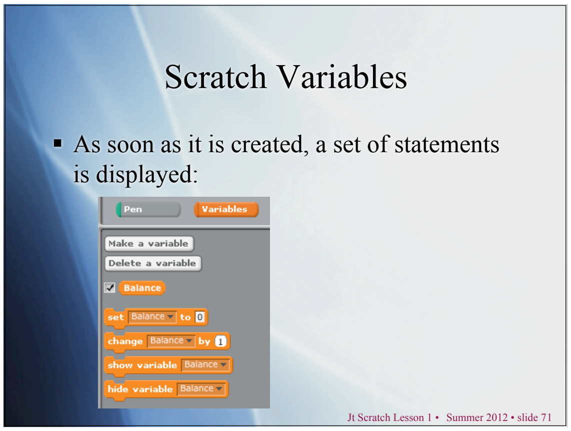
Balance

Check



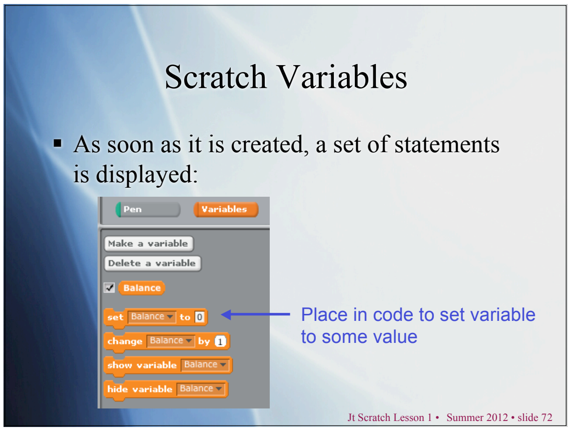
Scratch Variables

- Use Make a Variable



Scratch Variables

- As soon as it is created, a set of statements is displayed:



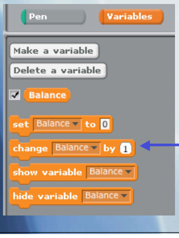
Scratch Variables

- As soon as it is created, a set of statements is displayed:

Place in code to set variable to some value

Scratch Variables

- As soon as it is created, a set of statements is displayed:

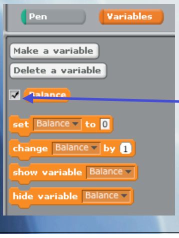


Place in code to change variable by a set amount

(such as increment)

Scratch Variables

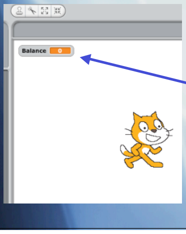
- As soon as it is created, a set of statements is displayed:



Check this box if you always want variable displayed on the stage

Scratch Variables

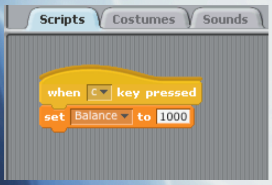
- As soon as it is created, a set of statements is displayed:



… if you always want variable displayed on the stage

Checkbook Code

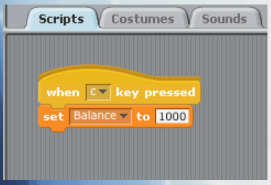
- So far:



```
when c key pressed
set Balance to 1000
```

Checkbook Code

- So far:

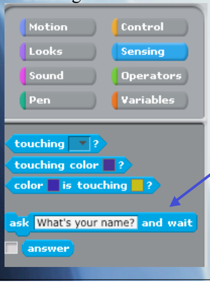


```
when c key pressed
set Balance to 1000
```

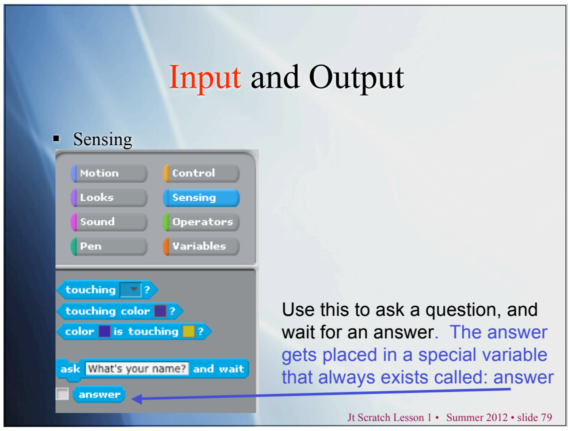
- How can we ask questions and display answers?

Input and Output

- Sensing



Use this to ask a question, and wait for an answer.

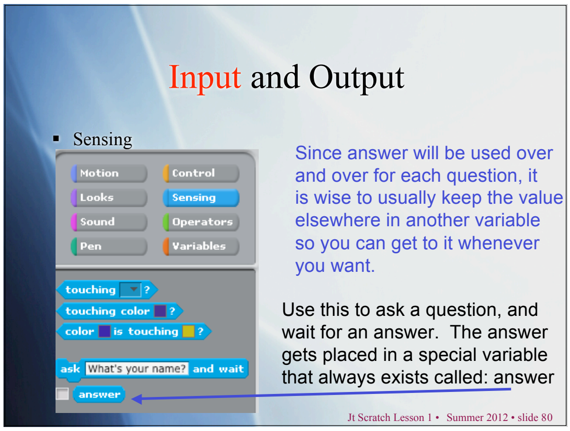


Input and Output

- Sensing

Use this to ask a question, and wait for an answer. The answer gets placed in a special variable that always exists called: answer

Jt Scratch Lesson 1 • Summer 2012 • slide 79



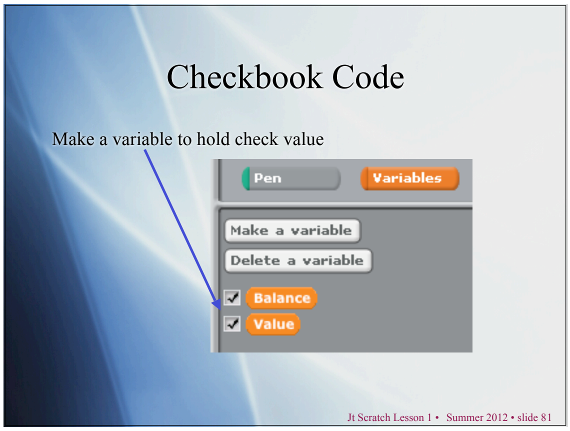
Input and Output

- Sensing

Since answer will be used over and over for each question, it is wise to usually keep the value elsewhere in another variable so you can get to it whenever you want.

Use this to ask a question, and wait for an answer. The answer gets placed in a special variable that always exists called: answer

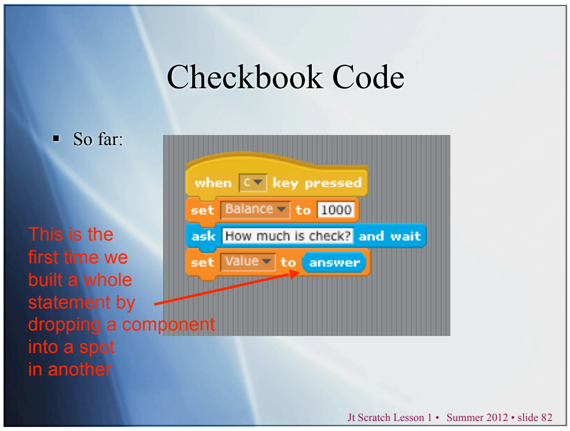
Jt Scratch Lesson 1 • Summer 2012 • slide 80



Checkbook Code

Make a variable to hold check value

Jt Scratch Lesson 1 • Summer 2012 • slide 81



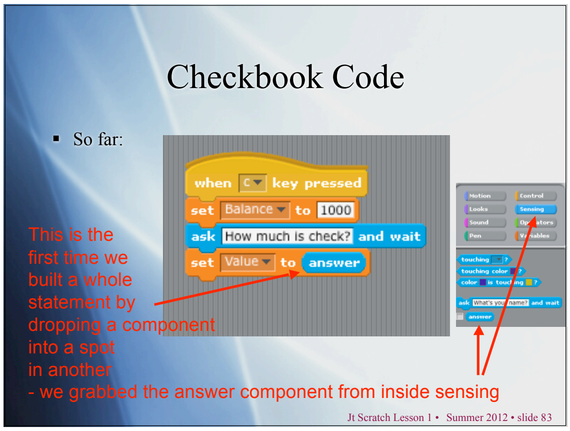
Checkbook Code

- So far:

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
```

This is the first time we built a whole statement by dropping a component into a spot in another

Jt Scratch Lesson 1 • Summer 2012 • slide 82



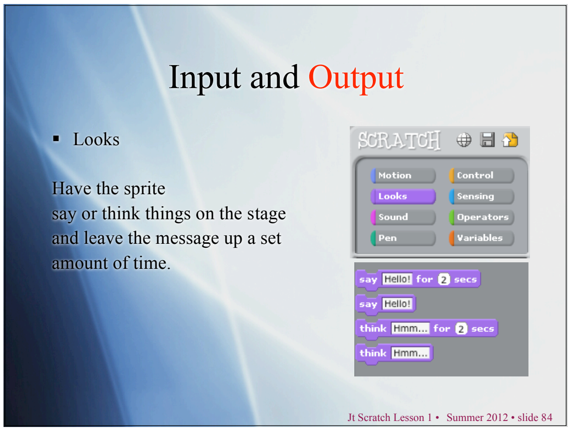
Checkbook Code

- So far:

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
```

This is the first time we built a whole statement by dropping a component into a spot in another
- we grabbed the answer component from inside sensing

Jt Scratch Lesson 1 • Summer 2012 • slide 83

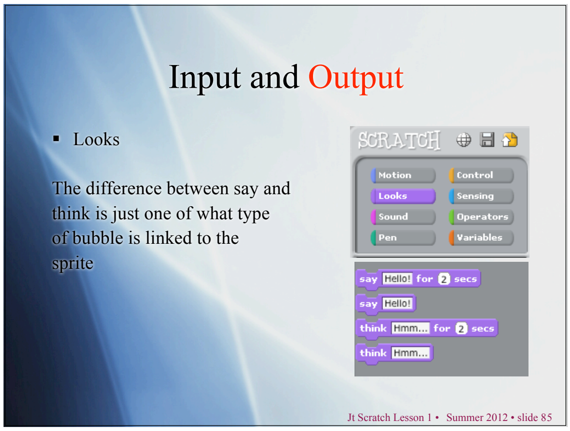


Input and Output

- Looks

Have the sprite say or think things on the stage and leave the message up a set amount of time.

Jt Scratch Lesson 1 • Summer 2012 • slide 84

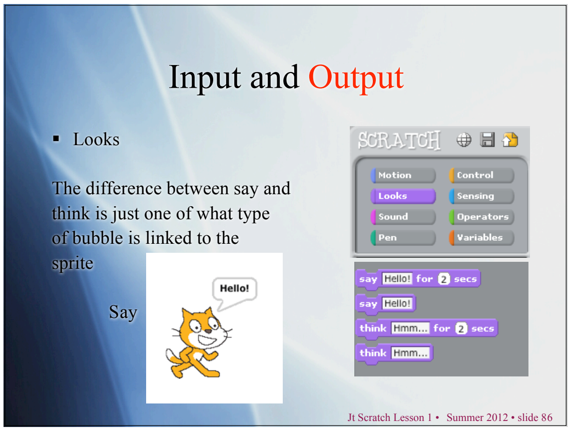


Input and Output

- Looks

The difference between say and think is just one of what type of bubble is linked to the sprite

```
say Hello! for 2 secs
say Hello!
think Hmm... for 2 secs
think Hmm...
```



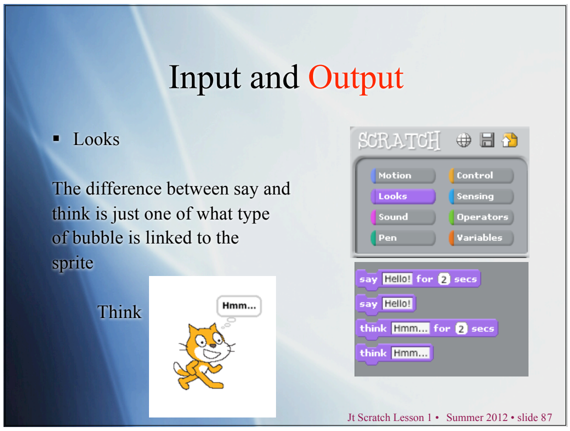
Input and Output

- Looks

The difference between say and think is just one of what type of bubble is linked to the sprite

Say

```
say Hello! for 2 secs
say Hello!
think Hmm... for 2 secs
think Hmm...
```



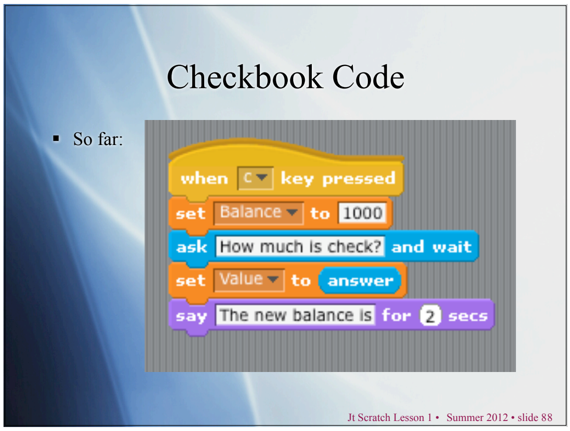
Input and Output

- Looks

The difference between say and think is just one of what type of bubble is linked to the sprite

Think

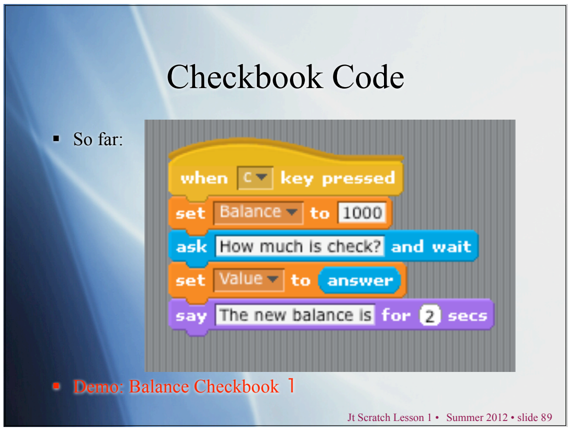
```
say Hello! for 2 secs
say Hello!
think Hmm... for 2 secs
think Hmm...
```



Checkbook Code

- So far:

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
say The new balance is for 2 secs
```

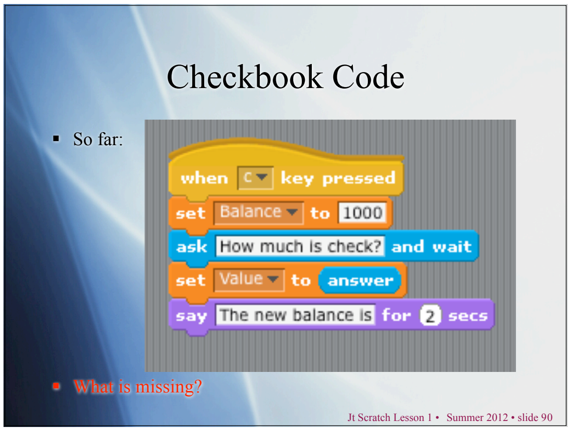


Checkbook Code

- So far:

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
say The new balance is for 2 secs
```

- Demo: Balance Checkbook 1



Checkbook Code

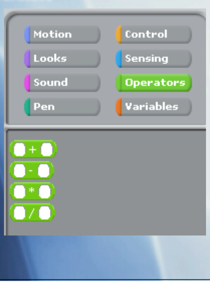
- So far:

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
say The new balance is for 2 secs
```

- What is missing?

Calculations

- Operators



Calculations

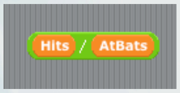
- Operators

Baseball: BattingAverage is Hits /Albats

Calculations

- Operators

Baseball: BattingAverage is Hits /Albats

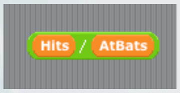


Hits / AtBats

Calculations

- Operators

Baseball: BattingAverage is Hits /Albats



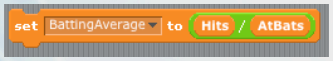
Hits / AtBats

- Now make this the value for BattingAverage

Calculations

- Operators

Baseball: BattingAverage is Hits /Albats



set BattingAverage to Hits / AtBats

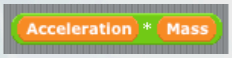
Now make this the value for BattingAverage

Drop this operator into Set Variable statement

Calculations

- Operators

Physics: Force is Mass * Acceleration

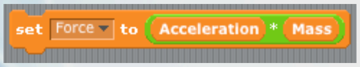


Acceleration * Mass

Calculations

- Operators

Physics: Force is Mass * Acceleration

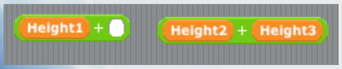


```
set Force to (Acceleration * Mass)
```

Calculations

- Operators

Operators can be combined together.
Figure the average of three heights.



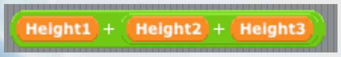
```
(Height1 + ( ))    (Height2 + Height3)
```

Build two addition operators. Slide one into the other.

Calculations

- Operators

Operators can be combined together.
Figure the average of three heights.



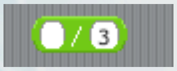
```
(Height1 + (Height2 + Height3))
```

Build two addition operators. Slide one into the other.

Calculations

- Operators

Operators can be combined together.
Figure the average of three heights.



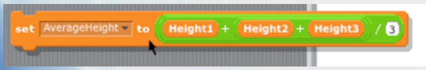
```
(( ) / 3)
```

Create a division operator.

Calculations

- Operators

Operators can be combined together.
Figure the average of three heights.



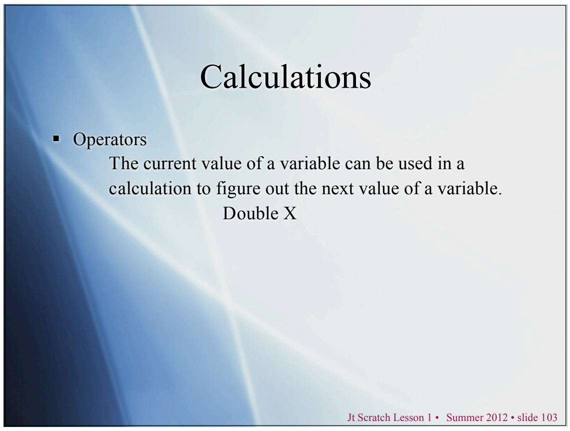
```
set AverageHeight to ((Height1 + (Height2 + Height3)) / 3)
```

Slide the three-way addition into the division.
Set this average variable to this complete calculation.

Calculations

- Operators

The current value of a variable can be used in a calculation to figure out the next value of a variable.

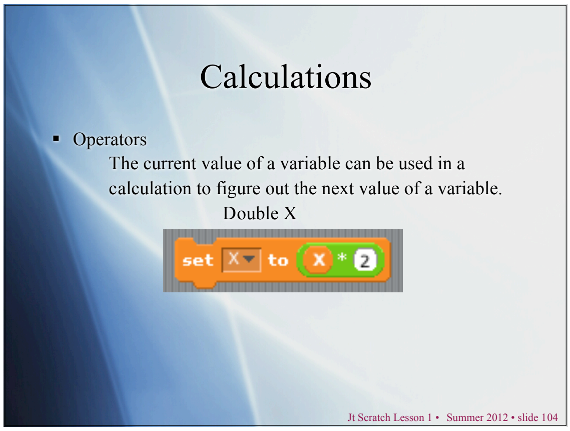


Calculations

- Operators

  The current value of a variable can be used in a calculation to figure out the next value of a variable.

  Double X



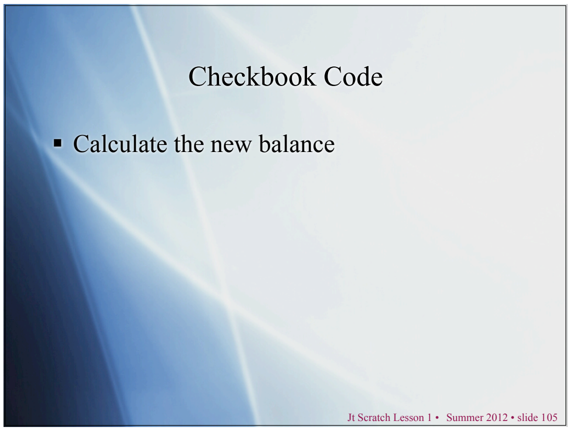
Calculations

- Operators

  The current value of a variable can be used in a calculation to figure out the next value of a variable.

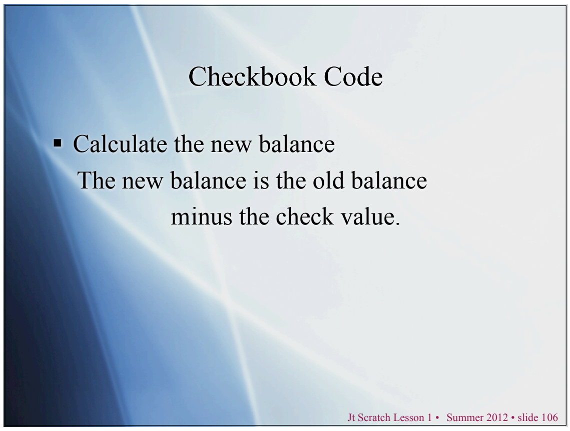
  Double X

```
set X to (X * 2)
```



Checkbook Code

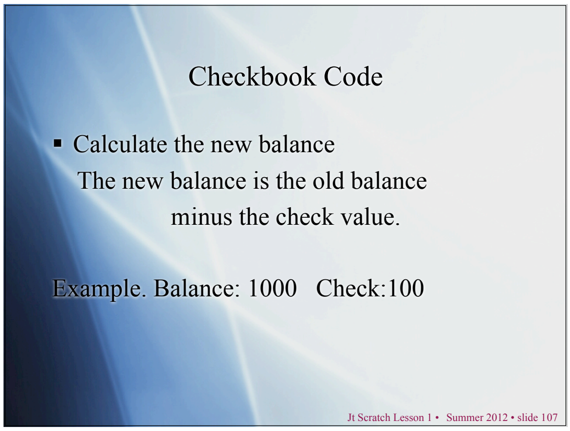
- Calculate the new balance



Checkbook Code

- Calculate the new balance

  The new balance is the old balance minus the check value.

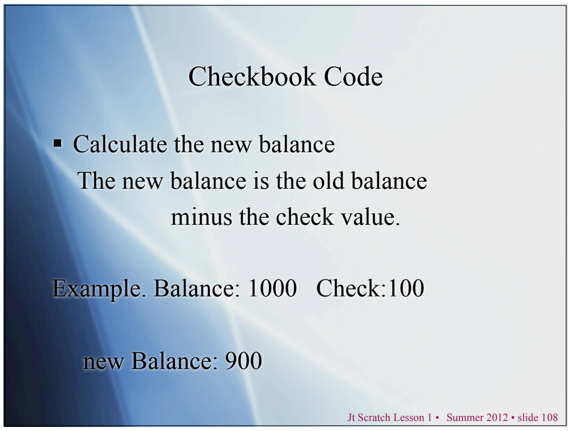


Checkbook Code

- Calculate the new balance

  The new balance is the old balance minus the check value.

Example. Balance: 1000 Check:100



Checkbook Code

- Calculate the new balance

  The new balance is the old balance minus the check value.

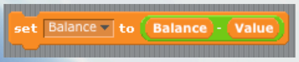
Example. Balance: 1000 Check:100

new Balance: 900

Checkbook Code

- Calculate the new balance

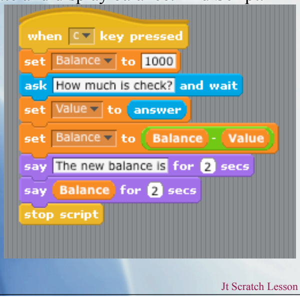
The new balance is the old balance minus the check value.



```
set Balance to (Balance - Value)
```

Checkbook Code

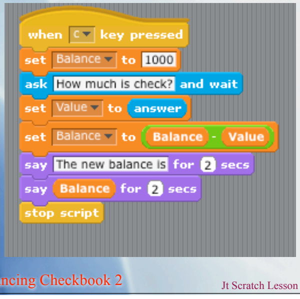
- Calculate and display balance. End Script.



```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say The new balance is for 2 secs
say Balance for 2 secs
stop script
```

Checkbook Code

- Calculate and display balance. End Script.

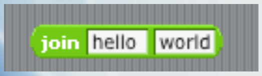


```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say The new balance is for 2 secs
say Balance for 2 secs
stop script
```

Demo: Balancing Checkbook 2

Other Operations

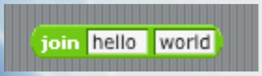
- Combining Text



```
join hello  world
```

Other Operations

- Combining Text

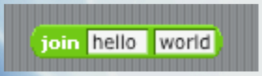


```
join hello  world
```

Notice space after hello?

Other Operations

- Combining Text

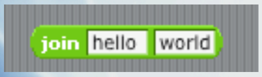


```
join hello  world
```

Notice space after hello?   This creates:

Other Operations

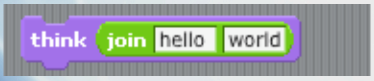
- Combining Text



Notice space after hello? This creates: hello world

Other Operations

- Combining Text



This can be given to Say or Think.

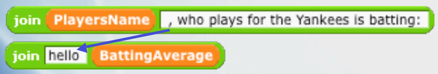
Other Operations

- Combining Text

Can combine a lot together - even text and variables -
to make more interesting statements.

Other Operations

- Combining Text

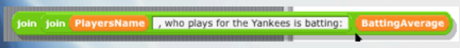


Can combine a lot together - even text and variables -
to make more interesting statements.

Place one join in another

Other Operations

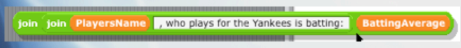
- Combining Text



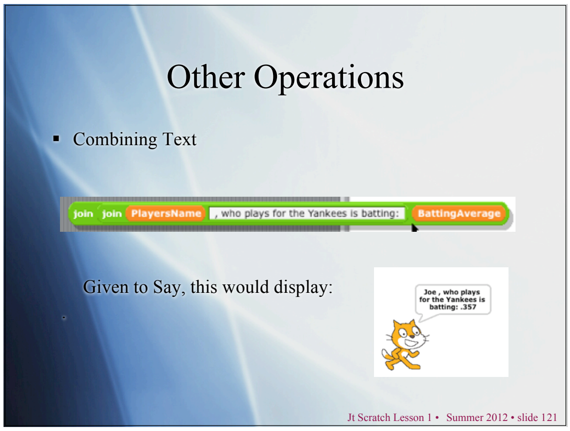
Can combine a lot together - even text and variables -
to make more interesting statements.

Other Operations

- Combining Text



Given to Say, this would display:



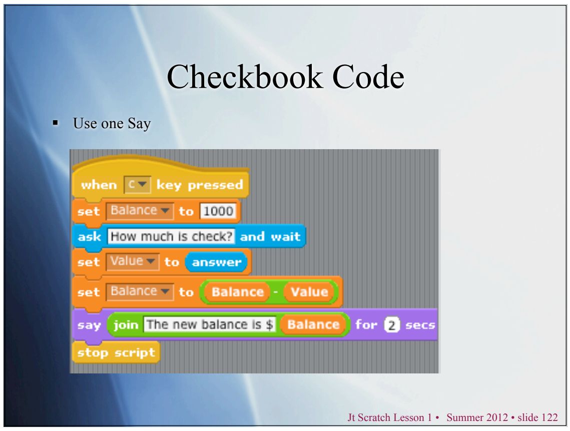
Other Operations

- Combining Text

```
join join PlayersName , who plays for the Yankees is batting: BattingAverage
```

Given to Say, this would display:

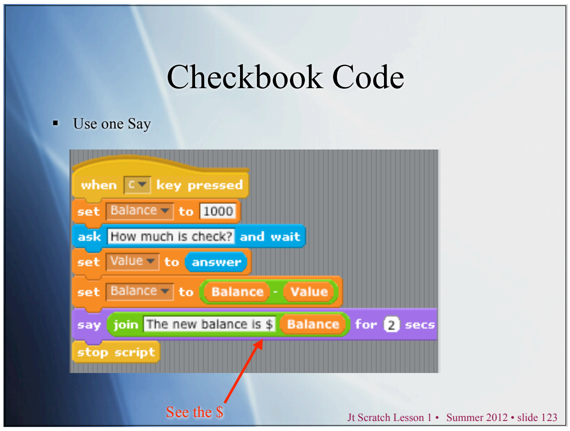
Joe , who plays for the Yankees is batting: .357



Checkbook Code

- Use one Say

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say join The new balance is $ Balance for 2 secs
stop script
```

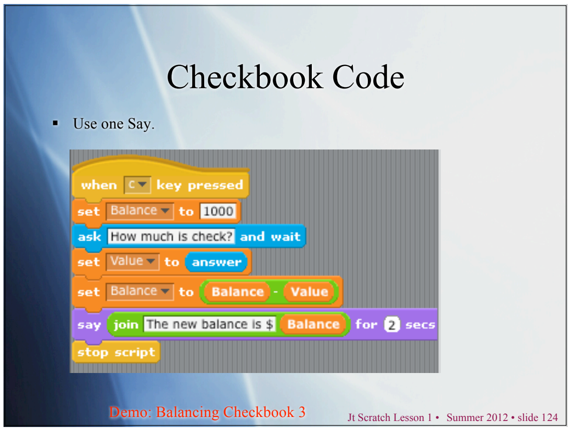


Checkbook Code

- Use one Say

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say join The new balance is $ Balance for 2 secs
stop script
```

See the $

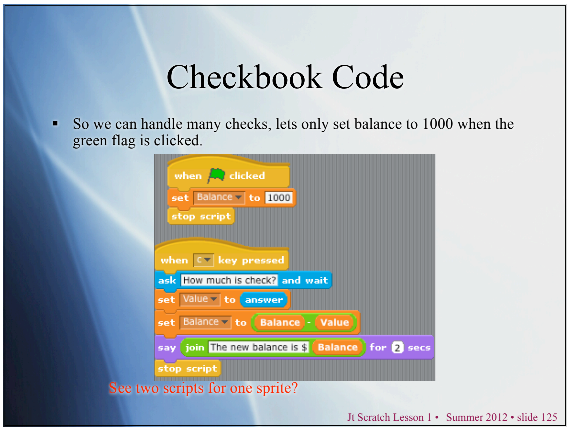


Checkbook Code

- Use one Say.

```
when c key pressed
set Balance to 1000
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say join The new balance is $ Balance for 2 secs
stop script
```

Demo: Balancing Checkbook 3



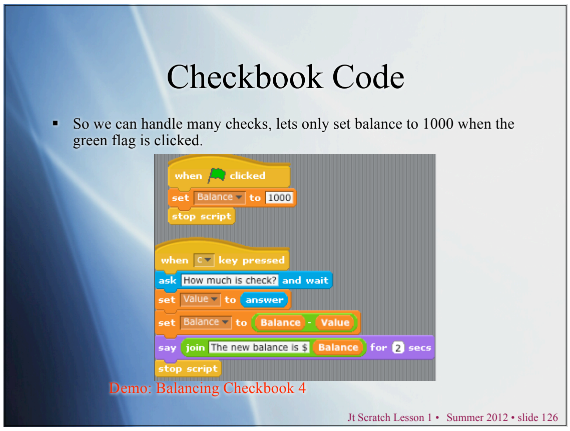
Checkbook Code

- So we can handle many checks, lets only set balance to 1000 when the green flag is clicked.

```
when clicked
set Balance to 1000
stop script

when c key pressed
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say join The new balance is $ Balance for 2 secs
stop script
```

See two scripts for one sprite?



Checkbook Code

- So we can handle many checks, lets only set balance to 1000 when the green flag is clicked.

```
when clicked
set Balance to 1000
stop script

when c key pressed
ask How much is check? and wait
set Value to answer
set Balance to (Balance - Value)
say join The new balance is $ Balance for 2 secs
stop script
```

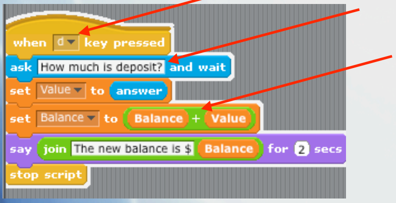
Demo: Balancing Checkbook 4

Checkbook Code

- Extension:
  Hey, how about handling deposits when d is clicked?

Checkbook Code

- Extension:
  Hey, how about handling deposits when d is clicked?



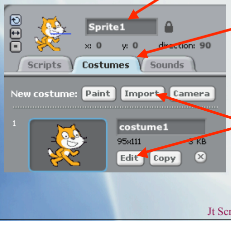
```
when [d] key pressed
ask [How much is deposit?] and wait
set [Value] to (answer)
set [Balance] to ((Balance) + (Value))
say (join [The new balance is $] (Balance)) for (2) secs
stop script
```

Demo: Balancing Checkbook 5

Checkbook Code

- Extension: Have some fun with multimedia.

Edit the sprite to be a checkbook:



Rename “checkbook”

Click costumes tab

Import or paint
a checkbook

Checkbook Code

- Extension: Have some fun with multimedia.

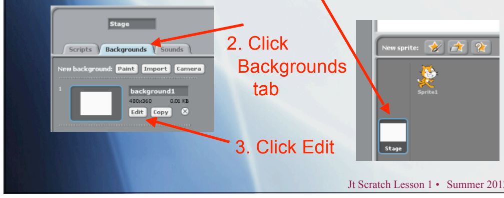
Edit the sprite to be a checkbook:

Checkbook Code

- Extension: Have some fun with multimedia.

Edit the stage to be a scale:

1. Click Stage



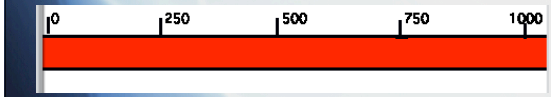
2. Click
Backgrounds
tab

3. Click Edit

Checkbook Code

- Extension: Have some fun with multimedia.

Edit the stage to be a scale:



Checkbook Code

- Extension: Have some fun with multimedia.
  Now include Motion commands in code to move
  checkbook sprite along the scale as the balance changes.

  - Go to X:? Y:?
    Place sprite at certain point on screen
    Hint: move pointer on stage and lower right
    will display x,y coordinates
  - Point in direction ?
    Specify direction you want sprite to point
    Next time it moves will be in that direction.
    -90 is left; 90 is right
  - Move ? Steps
    Move a certain amount of steps in current direction.

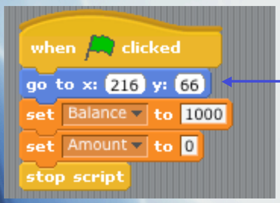
Checkbook Code

Full code.

Demo: Balancing Checkbook

Checkbook Code

Full code.

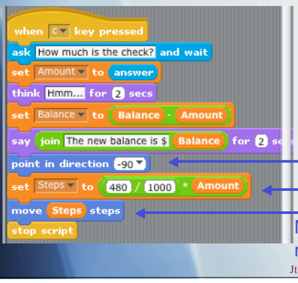


```
when [green flag] clicked
go to x: 216 y: 66
set Balance to 1000
set Amount to 0
stop script
```

Position sprite by 1000

Checkbook Code

Full code.



```
when c key pressed
ask How much is the check? and wait
set Amount to answer
think Hmm... for 2 secs
set Balance to (Balance - Amount)
say join The new balance is $ Balance for 2 secs
point in direction -90
set Steps to (480 / 1000 * Amount)
move Steps steps
stop script
```

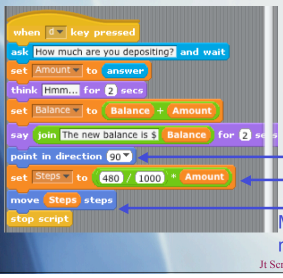
Face left

Calculate # of steps

Move that many steps

Checkbook Code

Full code.



```
when d key pressed
ask How much are you depositing? and wait
set Amount to answer
think Hmm... for 2 secs
set Balance to (Balance + Amount)
say join The new balance is $ Balance for 2 secs
point in direction 90
set Steps to (480 / 1000 * Amount)
move Steps steps
stop script
```

Point right

Calculate # of steps

Move that many steps

Availability of Slides

Go to

nbcs.rutgers.edu/~jt

to see the powerpoint slides and/or podcasts for this lecture