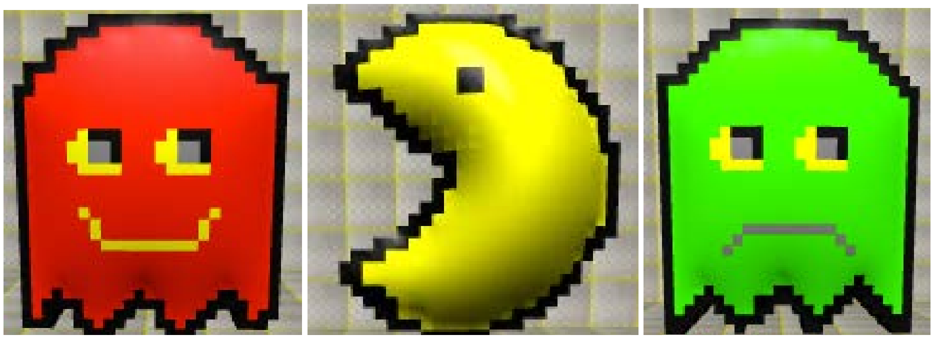


Creating “PacMan” With AgentCubes Online

Create the quintessential arcade game of the 80’s!
Wind your way through a maze while eating pellets.
Watch out for the ghosts!

Created by: Jeffrey Bush and Cathy Brand
University of Colorado, School of Education

This curriculum has been designed as part of the Scalable Games Design project.
It was created using portions of prior work completed by Susan Miller

This material is based upon work supported by the National Science Foundation under Grant No. DRL-1312129 and CNS-1138526. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Vocabulary/Definitions

Algorithm............a set of instructions designed to perform a specific task.

Attribute ....a variable belonging to an agent (such as scent) also called a Local Variable by computer scientists.

Brackets ....method of setting information apart using “[“ and “]”.

Broadcast .... controllers send out a message.

Ghost .... the agent that chases PacMan.

Collision ....an event wherein two agents run into each other.

Diffusion..............the process in which an attribute’s value (in this game, scent) is calculated based on the scent values of the neighboring agents.

Increment............to increase by one.

Hill Climbing.......a specific form of searching/seeking technique, or algorithm, by which the seeking/searching agent uses information (the value of the scent agent attribute) embedded in the floor agent.

Method ....a named set of rules evaluated by an agent in response to a message.

PacMan ....the main character who eats the pellets as the user moves him around the world.

Parentheses..........method of setting information apart using (and).

Polling ....the process of asking agents to update a simulation property and then taking some action based on the value of the simulation property.

Propagate............the spreading of the scent.

Randomly............to occur in non-predictable ways.

Rule Order..........the order in which rules are placed for each agent.

Simulation Property ....A named value that all agents can see and update.

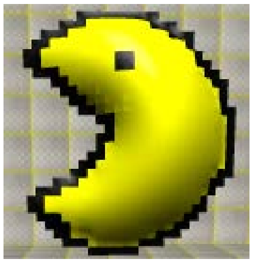
Student Handout 1A:

Part I - Basic Game

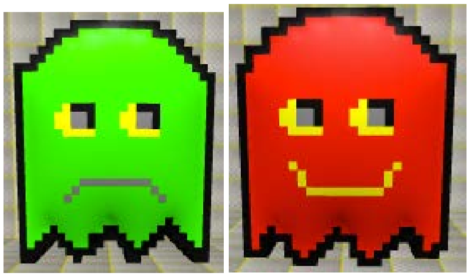
Initial Story: Create the quintessential arcade game of the 80's! Wind your way through a maze while eating pellets. Watch out for the ghosts!

Create these Agents and the world or clone our blank game with the agents already designed and start at step 11:

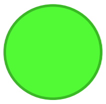
https://www.agentcubesonline.com/project/68900



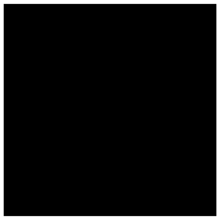
PacMan



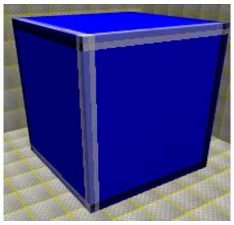
Ghost with two depictions



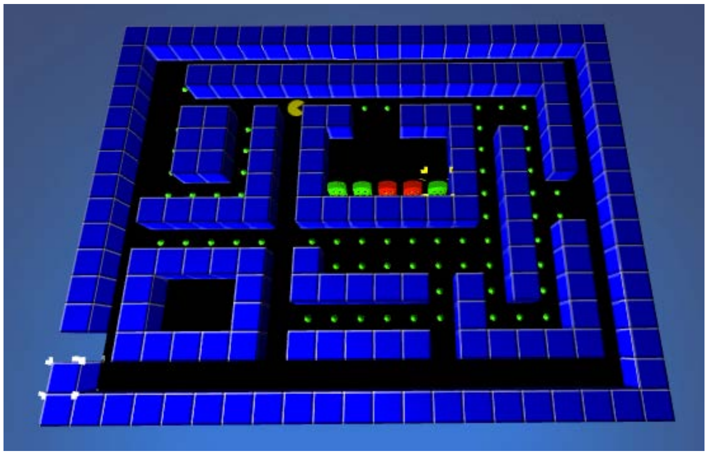
Pellet



Ground



Wall

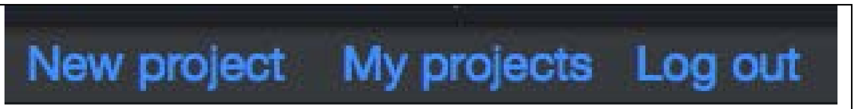
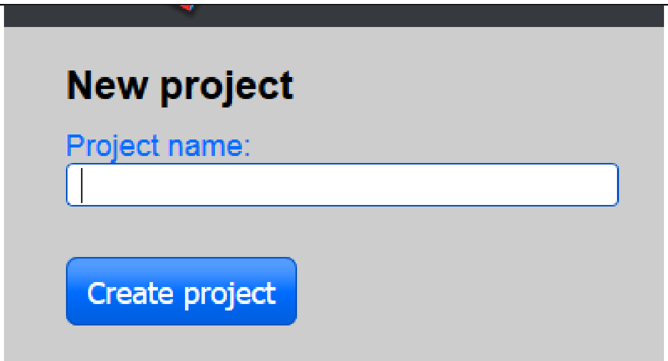


16x21 World with PacMan, Red & Green Ghosts

You are about to create your first game. Because this is new to you, we will give you some parts of the code. Be careful – you will have to find the patterns to create the rest of the code. If you get stuck, check in with a nearby student to get some help!

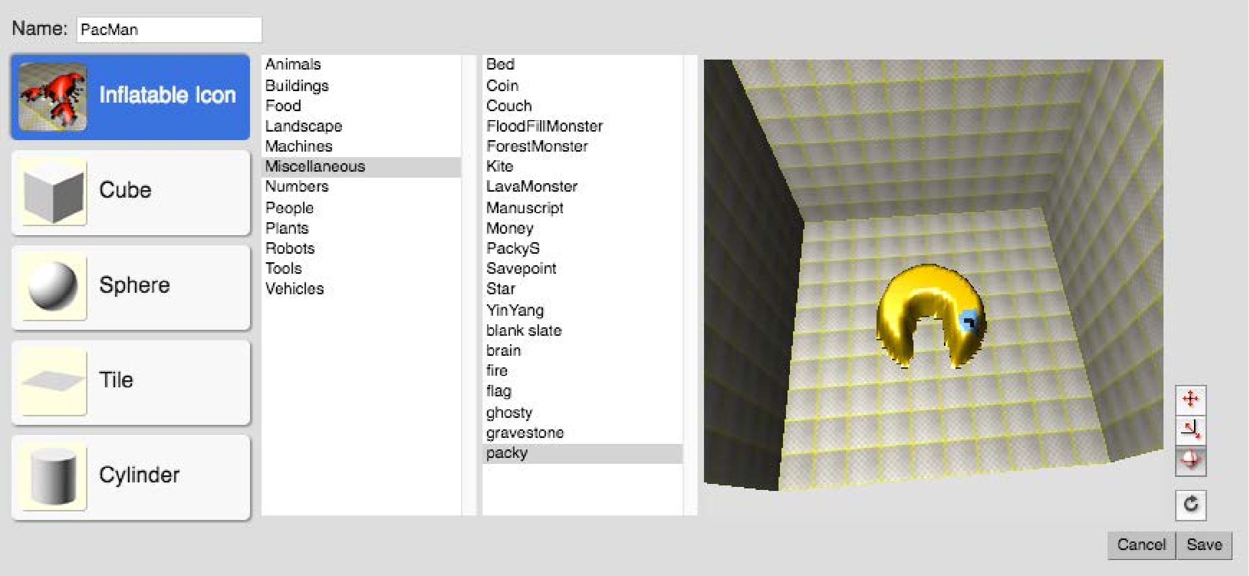
PacMan (Continued)

Launch AgentCubes Online and Create a New Game

Step 1: **Create Project**	 **Go to https://www.agentcubesonline.com/** **Click on the Login link.** **After you login, click on the green "New project" link below your login name.**
Step 2: **Name the Project** **Name it PacMan and click Create project.**	

Create the agents for PacMan

Step 3



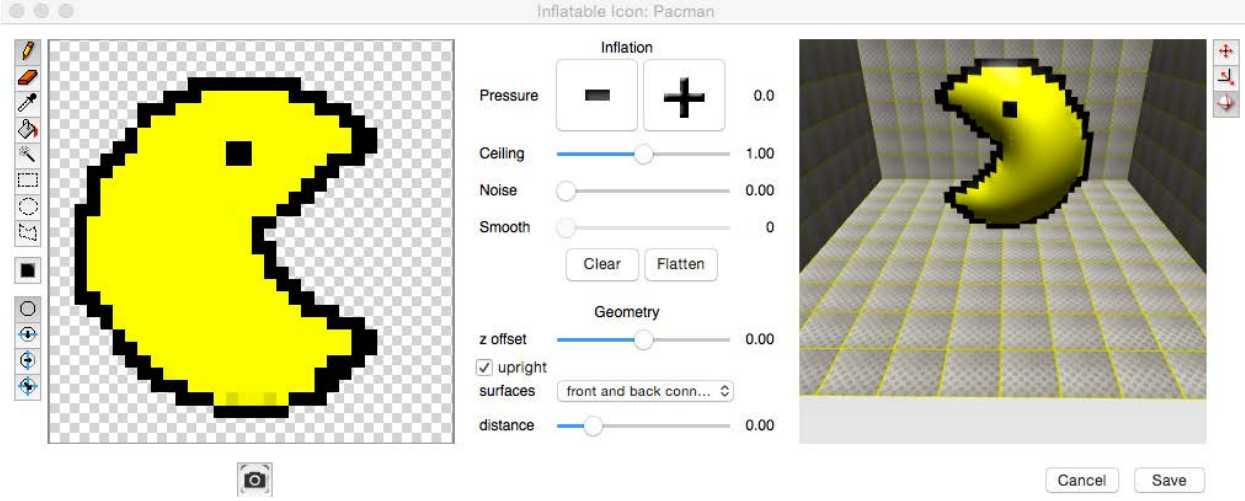
Name: PacMan

Click on the +Agent button.
Choose Inflatable Icon, then Miscellaneous, then "packy" to get a picture of PacMan.

Type PacMan in the Agent Name box and click on the OK button.

PacMan (Continued)

Step 4



Pressure 0.0

Ceiling 1.00

Noise 0.00

Smooth 0

z offset 0.00

☑ upright

surfaces front and back conn...

distance 0.00

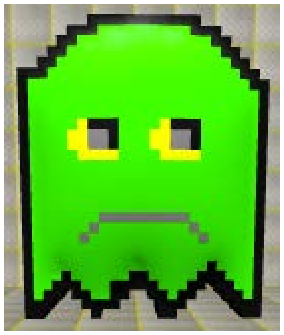
1. **To edit the drawing of PacMan, double click on the picture next to the agent's name.**
2. **Use the drawing tools to edit the image or click on the Clear button to erase the picture.**
3. **Use the pencil and the color picker (black square in the center of the row of drawing tools on the left of the 2D drawing) to draw a new picture.**
4. **Click on the color picker to choose a new color.**
5. **Use the bucket tool to fill in the outline with color.**
6. **Click on the + to inflate your drawing into a 3D image.**
7. **Use the Ceiling slider to flatten it.**
8. **Click the upright box to stand PacMan up.**
9. **Pick front and back connected next to the word "surfaces" so PacMan has 2 sides.**

Step 5

Create the Ghost.

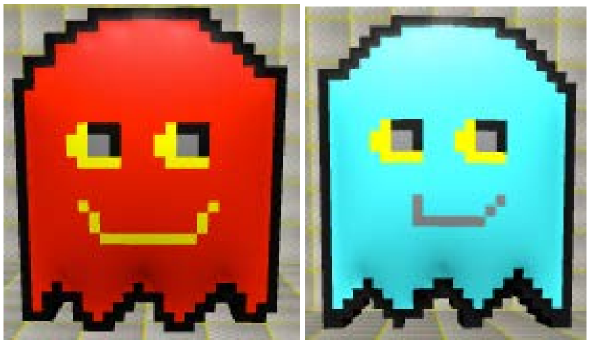
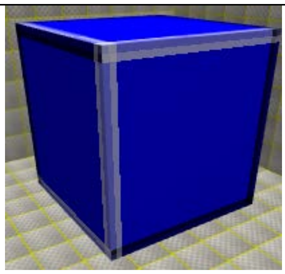
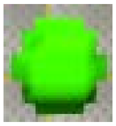
If you go to the Miscellaneous category under Inflatable Icon, there is an agent named "ghosty".

You can edit the drawing and change the color or draw your own ghost.

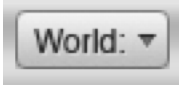
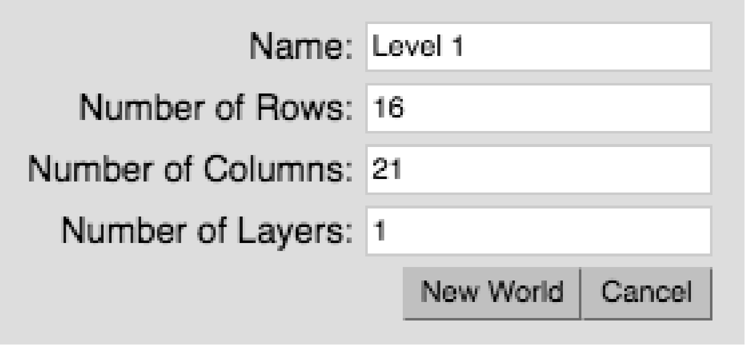
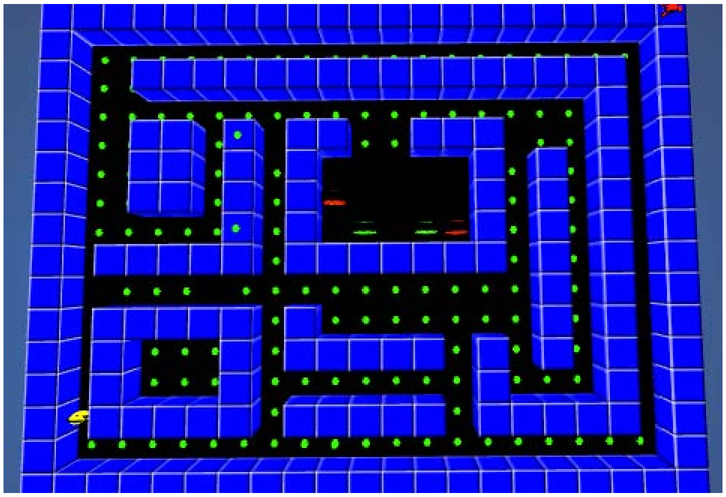
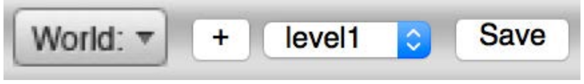


Here is a PacMan Ghost like the ones in the original game.

PacMan (Continued)

Step 6 **Make a second shape for your Ghost** **Click on the agent name, then click on the +Shape button at the bottom of the window.** **Give your new shape a name and click OK** **Then double click on the picture next to the name of the new shape and edit it.**	 **Change the ghost color so this shape looks different from the original ghost.**
Step 7 **Create the Wall** **Click +Agent and name it Wall** **Select Cube and then pick a cube that you like.**	 **Wall**
Step 8 **Create the Ground** **Click +Agent and name it Ground** **Select Tile and pick one that you like.**	 **Ground**
Step 9 **Create the Pellet** **Click +Agent and name it Pellet.**	 Pellet **Pick an inflatable icon, erase the picture, draw a pellet, use + to inflate & ceiling to flatten it.**

PacMan (Continued)

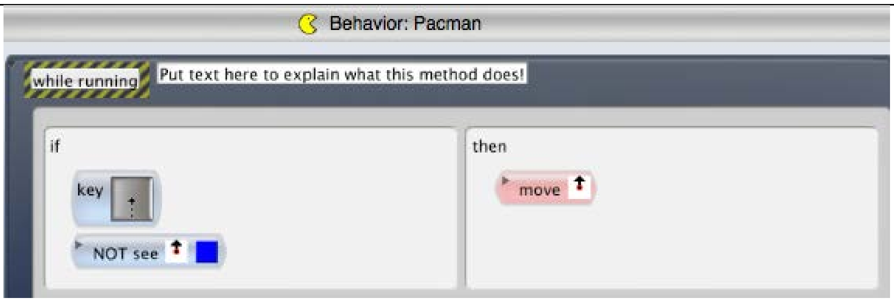
Step 10 **Make a Level 1 World** **Click on the 'down arrow' next to World** **When the New World box appears, type the name into the Name box.** **We recommend your world has at least 16 rows and 21 columns**	  Name: Level 1 Number of Rows: 16 Number of Columns: 21 Number of Layers: 1
Step 11 **Design your World** **Use the dotted rectangle tool to place a rectangle of walls on the World.** **Use the same tool to place the walls of the maze.** **Then place ground where there is not maze.** **Place a rectangle of pellets on top of the ground.** **Use the pencil to draw single agents (PacMan, the Ghosts).**	 The Ghosts are the in the rectangular area with an opening.
Step 12 **Save your World every time you make a change that you wish to keep.**	 **When agents get moved around after playing or testing, click on the Reset button on the top bar of AgentCubes Online to return to the saved version of your World!**

PacMan (Continued)

Time to Create Behaviors for your Agents

<table>
<tr>
<td><b>Step 13
Create an agent behavior</b>

<b>Click on an agent and its behavior window will appear below the world.</b>

<b>You are going to drop and drag the conditions (on the left) and the actions (on the right) to create the rules.</b></td>
<td>
Behavior: Pacman
while running
Put text here to explain what this method does!
if
key
NOT see
then
move

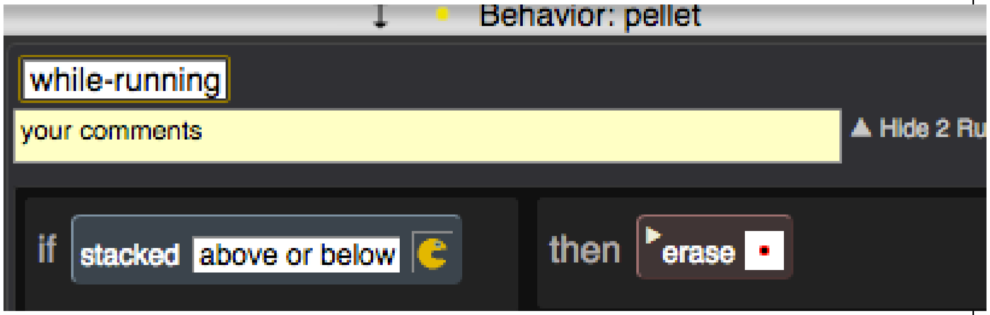
<b>This rule says:</b>

<b>IF the user presses the up arrow,
AND I Do NOT See a Wall in the up direction,
THEN my PacMan will move UP</b>

<b>Tip: Click on the See condition, then the NOT button at the bottom of the window to make the NOT see condition.</b>

<b>Create 3 more rules to make PacMan move right, left and down.</b>

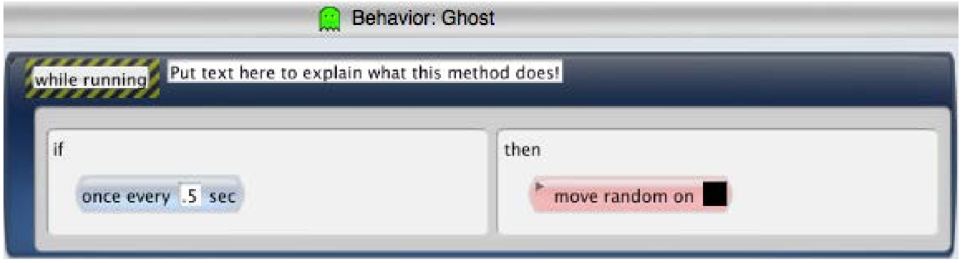
<b>NOTE: Click on the rule you made, then click on the +RULE button at the bottom of the window to create each new rule.</b></td>
</tr>
<tr>
<td><b>Step 14
Program the PacMan to eat the pellets.</b></td>
<td><b>First, take a moment to think about the PacMan and the Pellets When the PacMan collides with the Pellets, then the Pellet disappears, making it look like the PacMan ate it.</b>

<b>Take a look at this code when the Pellet sees the PacMan above or below him: It says…</b>

Behavior: pellet
while-running
your comments
▲ Hide 2 Ru
if
stacked
above or below
then
erase

<b>IF I see a PacMan above or below me</b>

<b>THEN... erase myself (the pellet)</b></td>
</tr>
</table>

PacMan (Continued)

	Play with adding a wait time before erase. What happens when wait before adding the pellet? Choose a wait time (or no wait time) that you think makes the eating behavior look the way you want it.
Step 15 **Program the Ghosts to move randomly**	**Click on the agent to add behaviors to that agent.**  Behavior: Ghost while running — Put text here to explain what this method does! if: once every .5 sec then: move random on ■ ***How can you make the Ghost move faster or slower? Experiment with the number in the <u>once every</u> condition.*** ***The ghost should also move on the pellets so add a second rule!*** Be mindful of what will happen if both commands have the same time for "once every". (see Appendix I: <u>Guidance on Ghost's Random Movement</u> for more help)
Step 16 **Create rule to end the game when the <u>PacMan</u> is next to the Ghost** **Click on PacMan and add this rule.** **<u>Note: You need a similar rule for each different Ghost shape.</u>**	This second Ghost rule says: IF I am <u>NEXT TO at least 1 PacMan</u> THEN, stop the simulation, <u>show a message</u> "You lose!" AND <u>Reload the World</u> **Very Important Tip: Put** stop simulation **and/or** reload world **in your Game ending rule.** If you forget to do this, **AgentCubes Online will do the game ending rule over and over** until you are able to type the Return Key to click the OK on the dialog box and then immediately after use the mouse to click on the stop game button ▶ ■ ▮▶ (the red square). If you cannot click on the stop game button before the dialog box appears, you must close the browser window, then open a new browser window, go to AgentCubes Online, find your project and click on the edit button so it opens.

Student Handout 2

Part 2 – Making the Ghost Chase the PacMan

So far, your Ghost just moves randomly, either just on the floor, or on the floor and the pellets…he doesn't actually chase the PacMan, does he? That's about to change!

The Ghost will intelligently seek the PacMan agent using a computational thinking pattern called "searching." In this instance, we will use a specific method of searching called Hill Climbing. Imagine the PacMan agent emits a scent. Hill climbing is a procedure or **algorithm** to find the direction in which the scent is strongest.

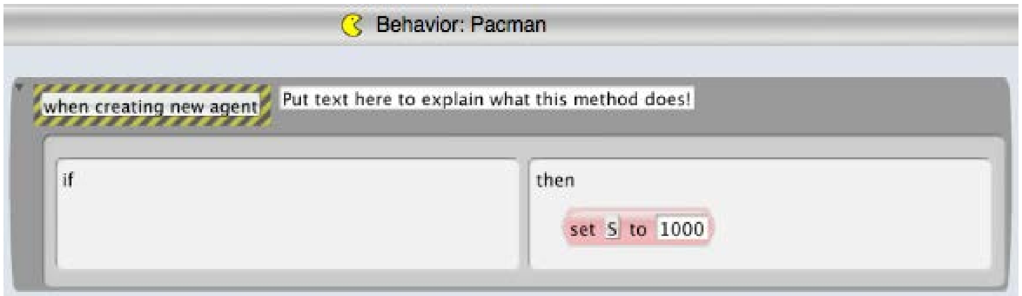
The scent will spread out, or be **propagated,** by the ground agents using a computational thinking pattern called "diffusion." Diffusion is a fundamental process (physical, biological, and social) by which objects move from areas of highest concentration to areas of lowest concentrations. The closer to the source of the scent, the greater its value[1].

This phase of the project introduces the concept of an "**agent attribute**," which is unique information that is stored within each occurrence of an agent. Computer scientists call this agent attribute a **local variable**.

Step 1: The best way to initialize PacMan's S agent attribute is to set it when PacMan is drawn on the world because then PacMan s attribute will always start at the same value.

To do this, create a new Method by clicking on the +Method button. Click on the word "on" in the new method's black and yellow striped tape and change the label from "on" to "when-creating-new-agent".

Your when-creating-new-agent method should look as follows:



Behavior: Pacman

when creating new agent — Put text here to explain what this method does!

if	then
	set S to 1000

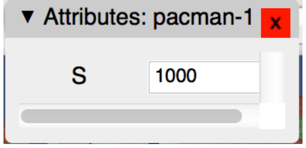
If you use this method to set PacMan's S attribute, make sure that you erase and redraw PacMan and then **SAVE** the World.

PacMan (Continued)

Important Note: If you forget to save the world, PacMan may not have any value set for S when the world is reloaded.

Checking the Value of PacMan's S agent attribute:

1. Double click on PacMan with the big arrow tool .
2. You should see this window appear:



▼ Attributes: pacman-1

S 1000

3. If S is not visible in the window, it did not get set to a value yet.
4. Erase and redraw PacMan and then **save** the world. Then S should appear in the Attributes window.

Step 2:
Now, since the scent is diffusing, or spreading out, we need to find the average of the scent from the area around a ground agent or a pellet agent. Think of it as the smells are coming in from the North, South, East and West. The smell in the center, then, is the average of these four smells. How will you create that programmatically?

Why do we multiply by 0.25?

When you find the average of a set of numbers, you add them up and divide by the number of numbers.

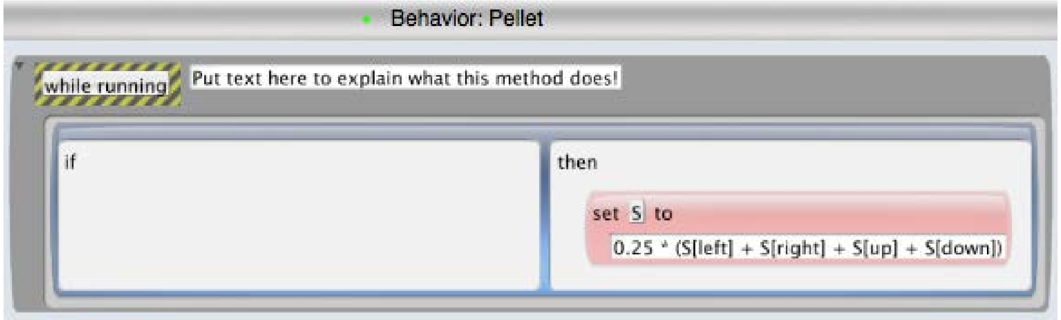
In this case, dividing by 4 is the same as multiplying by ¼ which equals 0.25

Diffuse the scent using the pellet agents

The pellet agent will have the behavior below; the single action is to calculate and store the average of the four surrounding agents' agent attributes. Remember, you named the agent attribute "S" (for scent).

The "set" action sets each pellet agent's attribute "S" to the <u>average</u> of the attributes in the agents above, below, and on each side:

$$S = 0.25*(s[up]+s[down]+s[right]+s[left])$$



Behavior: Pellet

while running — Put text here to explain what this method does!

if

then

set S to 0.25 * (S[left] + S[right] + S[up] + S[down])

NOW…diffuse the scent across the ground by adding a rule to the ground agents!

Match both the parentheses "(" and the brackets "[" as shown in the equation.

Step 3: For the Ghost to know which way to walk, he has to determine where the scent is the strongest. We call this HILL CLIMBING. If this were real life, he would smell up, smell down, smell left and smell right. Wherever the smell was strongest, he would walk in that direction. We need to program the Ghost to do this.

What do FIRE ALARMS have to do with coding?
A METHOD is a set of rules with a name…rules to follow in a specific situation. These are done when there is a specific call for them…much like the fire alarm means you follow different rules. You can create a METHOD by clicking the +Method button below an agent's

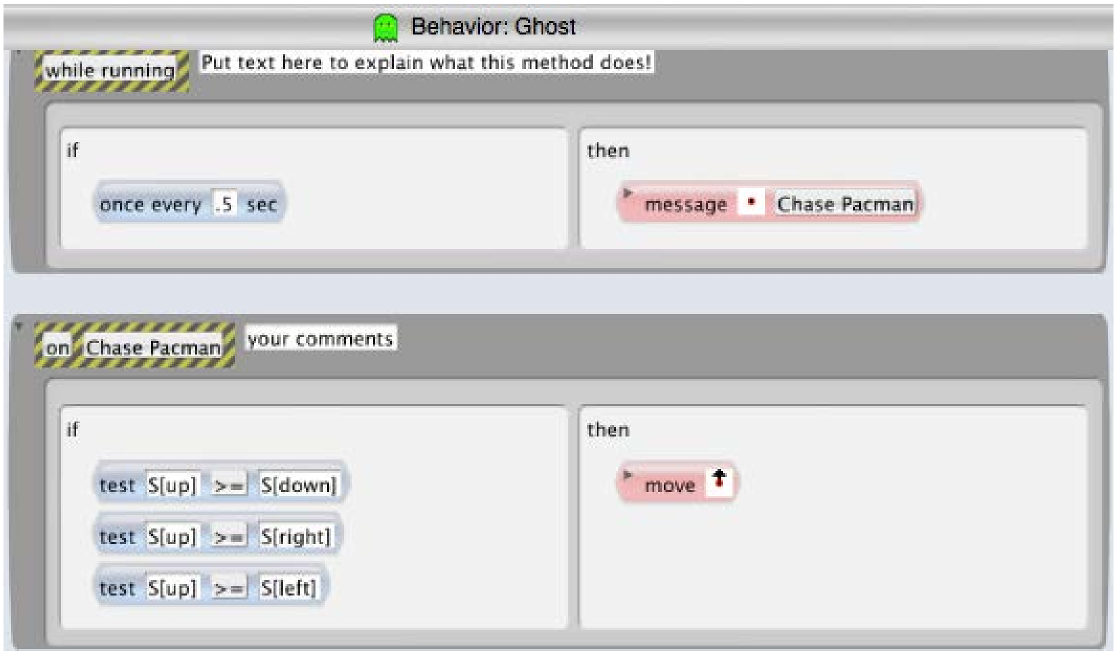
We will create a METHOD for the Ghost to follow a set of rules.

Take a look at the programming below.

The rule in the while running method says
"ONCE EVERY 0.5 seconds, follow the Chase PacMan procedure".

The rule in the Chase PacMan method says
"IF the smell **above you** is greater than or equal to any of the other smells in different directions (down, left or right), THEN move up."

PacMan (Continued)



```
Behavior: Ghost

while running   Put text here to explain what this method does!
  if                          then
    once every .5 sec           message • Chase Pacman

on Chase Pacman   your comments
  if                          then
    test S[up] >= S[down]       move ↑
    test S[up] >= S[right]
    test S[up] >= S[left]
```

Now, add the three more rules to the Chase Pacman method so that the Ghost knows what to do if the smell down (S[down]) is greater. What if the smell to the left is greater? What about the smell to the right?

Run your game to see if the Ghost chases the PacMan!
If it isn't working, check the following:

- Erase and redraw PacMan, then **save** the world so the initial value of S is saved.
- In the Chaser's rules, the method name must be the same in the message action and the black and yellow striped method name tag of the hill climbing method!
- Use of parentheses "(" and brackets "[" in the ground and pellet agent rules must be correct. Check the picture of the ground agent's equation 2 pages ago and compare it to the equations in your ground and pellet agents.
- Check your hill climbing rules again and make sure that the arrows in the actions point the correct direction and that the conditions for each rule are correct.

Student Handout 3 Part 3:

Making the game more sophisticated – Polling and Broadcast

In this enhancement to the PacMan project, the PacMan must "eat" all of the pellets in order to win. Polling will tell us when all the pellets are gone and PacMan has won.

Polling uses a **simulation property**, also called a **global variable** by computer scientists, which is a piece of information that all agents in the simulation or game may check or set if they have the correct rules. A controller agent does the polling by sending out a message at intervals to all the agents that must be counted. These agents respond by adding one to the simulation property. The controller determines when all the pellet agents are gone and PacMan has won.

The teacher has given an assignment to the class and wants to know if everyone is finished. She says to the class, "Put your hand up if you are still working." Hands go up. She counts them – there are five students still working. "Okay, put your hands down and keep working."

A few minutes later, she does it again. She says to the class, "Put your hand up if you are still working." Hands go up. She counts them – there are two students still working. "Okay, put your hands down and keep working."

A few minutes later, she does it again. She says to the class, "Put your hand up if you are still working." This time, no hands go up. "Everyone is done, put your books away."

That's what this programming will look like.

The Controller will say, "Pellet count starts at zero" (like the classroom, no hands are up when the teacher asks who is still working).

When the pellets 'hear' the Controller ask (broadcast) the question, the pellets respond back (raise their hands).

Definition: Computer scientists call the process of making a decision by sending a message to multiple recipients and checking responses **polling**.

PacMan (Continued)

The Controller counts the pellets. If the answer is more than zero, nothing happens and the game continues. If the answer is zero (meaning that there are no remaining pellets on the board), the game ends.

Step 1: Create the Pellets simulation property as described in the green box below.

Step 2: Create the Controller agent.

- Use +Agent to make a Controller agent and choose any image.
- Place the Controller agent on top of a wall in your PacMan world and SAVE the world!

Step 3: Add a rule to the Controller agent's while running method.

1. Set the number of pellets to zero. (this is like the teacher saying "hands down")
   Set @Pellets to zero
2. Ask the pellets if they are still on the world
   Broadcast to Pellet agents to do "Count"
3. Check the number of pellets to see if the game is done.
   Send a Checkwin message to myself.

Step 4: Program the Controller agent's Checkwin method.

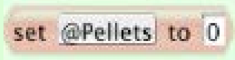
- If there are no pellets left, tell the player that PacMan won and stop the game.

Step 5: Program the Pellet Agent's Count method.

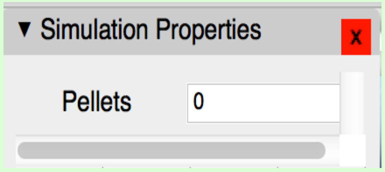
1. Make a new method for the Pellet agent.
2. Name it Count.
   The name must exactly match the name broadcast by the Controller.
3. Add a rule with an action that sets the value of "**@Pellets**" to "**@Pellets + 1**".
   This is how programmers add 1 to a number.

How do Simulation Properties Work?

1. Simulation properties may be added, saved or deleted in the Simulation Properties window.
2. Open this window by clicking on the gear button on the top right side of the AgentCubes Online window and choosing "Show Simulation Properties". ⚙▾
3. Make the Pellets simulation property by clicking on the + button at the bottom of the simulation properties window and typing the name "Pellets".
4. Click on the Save button so that the new simulation property is saved!
5. The value of a simulation property can be changed or checked by any of the agents in the game or world.
6. All conditions and actions which check or change the value of a simulation property must place an "@" before the simulation property name.



set @Pellets to 0



▼ Simulation Properties

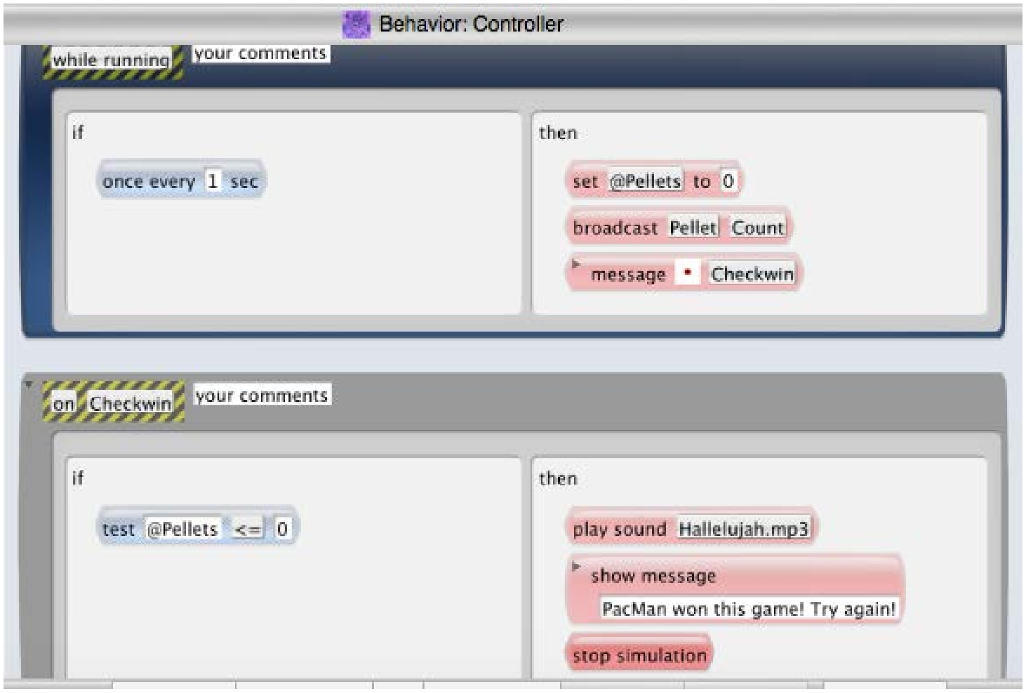
Pellets	0

Try setting up these rules now!

Count is not part of the continually running "While Running" method. It must be a separate method since it only runs when called by the controller agent.

Check your program:

Here is the Controller agent behavior with the rule in the while running method that makes the pellet agents count themselves and the new Checkwin method that ends the game if PacMan has eaten all the pellets:

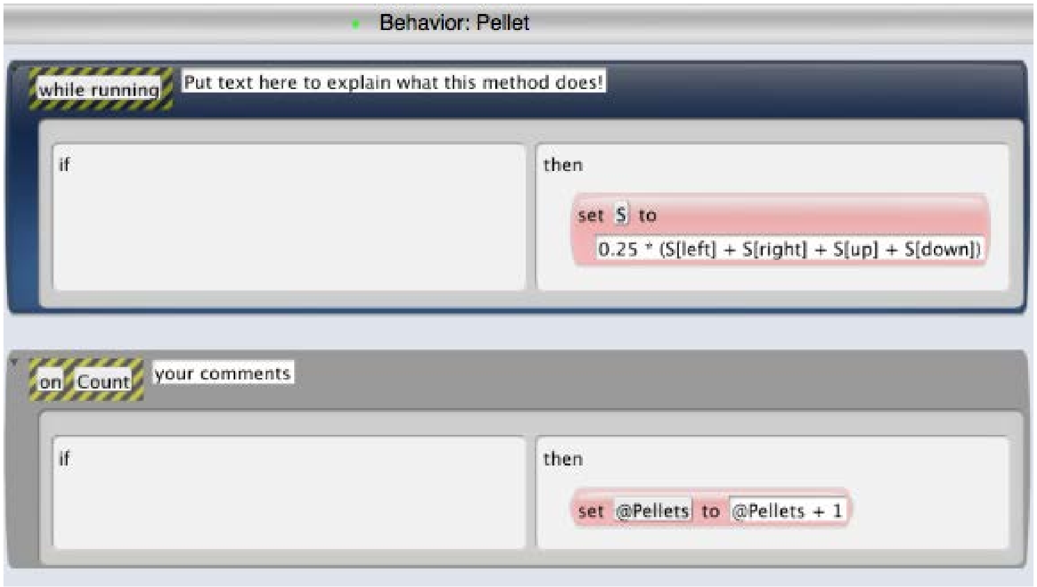


```
Behavior: Controller

while running    your comments
  if                       then
    once every 1 sec         set @Pellets to 0
                             broadcast Pellet Count
                             message • Checkwin

on Checkwin    your comments
  if                       then
    test @Pellets <= 0       play sound Hallelujah.mp3
                             show message
                               PacMan won this game! Try again!
                             stop simulation
```

Here is the Pellet behavior with the new Count method that allows each Pellet agent to add 1 to the Pellets simulation property:



```
Behavior: Pellet

while running    Put text here to explain what this method does!
  if                       then
                             set S to
                               0.25 * (S[left] + S[right] + S[up] + S[down])

on Count    your comments
  if                       then
                             set @Pellets to @Pellets + 1
```

Student Handout 4a:

Challenge 1:PacMan Changes Direction

Change Directions

Make the PacMan face the direction he's heading

Before your start this challenge:

You must have a complete basic PacMan game with a PacMan who wins if he eats all the pellets and Ghosts who either move randomly or chase the PacMan. The PacMan loses if a Ghost gets too close. The world should have walls that the Ghost and PacMan cannot cross.

Description of the Challenge:

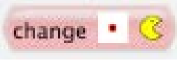
PacMan will turn in the direction he's heading.

Keep in mind there are multiple ways to solve this. Before reading ahead, try solving this challenge your own way. Do you need new/different rules? New agents/shapes?

Option 1: Each PacMan follows the same rules so you need 3 new shapes instead of a new agent

Steps:

- Select PacMan by clicking on him, then click on +Shape
- Clear the picture and draw PacMan facing a different direction.
- Use the change action with a dot in the middle because the means "change me to" so the agent is able to change its shape.



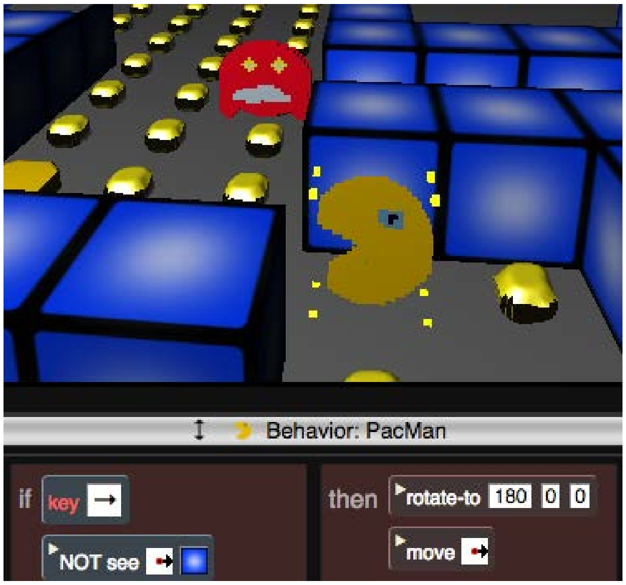
change

Option 2: Instead of making new shapes, use the "rotate to" command

Steps:

- Edit the move behavior of PacMan to include a "rotate to" command
- The first of the three numbers in "rotate to" is the rotation you want, a rotation of 180 degrees will cause PacMan to turn in the opposite direction.

Once you are done, TEST your program to confirm that the PacMan's shape changes when he changes directions as he moves.



Behavior: PacMan

if key → NOT see → ■ then rotate-to 180 0 0 move →

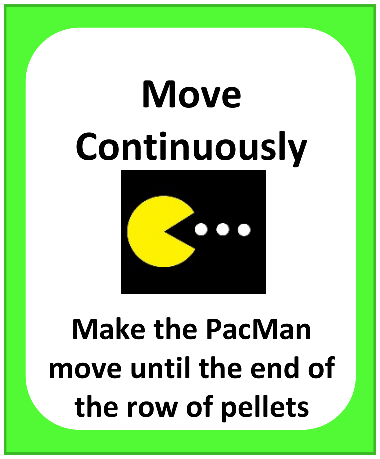
Student Handout 4b:

Challenge 2: PacMan Moves Continuously

Before your start this challenge:

You must have a complete basic PacMan game with a PacMan who wins if he eats all the pellets and Ghosts who either move randomly or chase the PacMan. The PacMan loses if a Ghost gets too close. The world should have walls that the Ghost and PacMan cannot cross.

You must have 4 different shapes for the PacMan so that he faces the direction he heads.



Move Continuously

Make the PacMan move until the end of the row of pellets

Description of the Challenge:

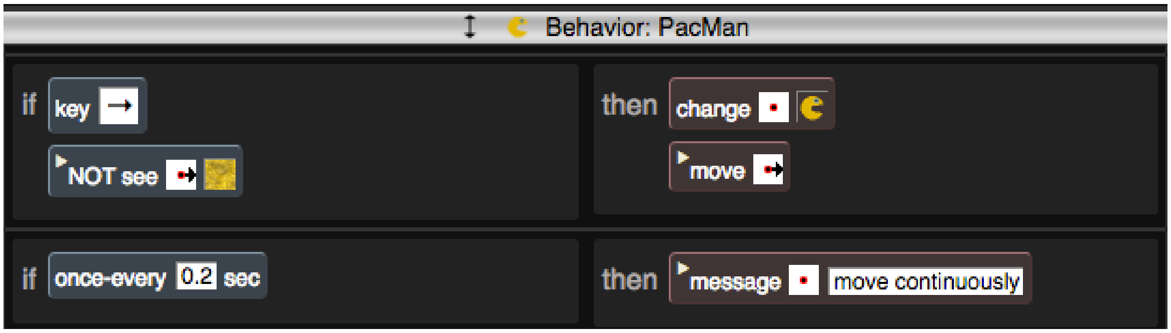
PacMan will continuously move in the direction he's heading.

This challenge gets you started, but won't give you all the code. Review the code below: It says, when the right arrow is pressed AND I do not see a wall to the right, change to the right-facing depiction. Once every 0.2 second, make me (the PacMan) do "move continuously".

When the move continuously method is called, the PacMan does the following:
If I see myself heading right AND I do not see a wall in the right direction, I will move right.

The effect of the rule in move continuously is to make PacMan keep moving whichever way he is facing as long as there are no walls in the way.

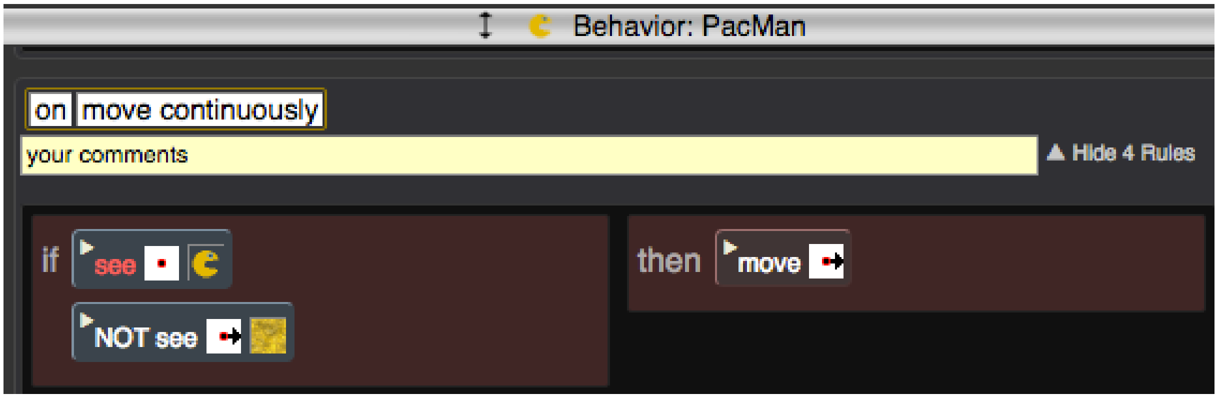
PacMan (Continued)



Behavior: PacMan

if key →
NOT see
then change
move

if once-every 0.2 sec
then message move continuously



Behavior: PacMan

on move continuously
your comments
▲ Hide 4 Rules

if see
NOT see
then move

There is still much to code:

Step 1:

Create code for all the other directions**.**

Step 2:

Test your program. (Hint: be sure your PacMan still leaves his scent everywhere.) **Click** on the PacMan with the big arrow tool to select him and run the program.

Use the colors to decide which rules are true or false. In this case, the first rule is red, which means the Right arrow was not pressed or a wall was in the way.

The next rule is green, which means every 0.2 seconds, the PacMan is being told to do MoveDirection.

The method MoveDirection is green, which means that either or both conditions are true. The PacMan does sees his right-facing shape AND does not see a wall, making the rule TRUE, so PacMan will move one step to the right.

Student Handout 4c:

Challenge 3: Power Pellet

Before your start this challenge:

You must have a complete basic PacMan game with a PacMan who wins if he eats all the pellets and Ghosts who either move randomly or chase the PacMan. The PacMan loses if a Ghost gets too close. The world should have walls that the Ghost and PacMan cannot cross.

You must have different depictions of the PacMan so that he faces the direction he heads, and he must move continuously.

Power Pellets

Make a Power Pellet that allows the PacMan to eat the ghosts.

Description of the Challenge:

- Power Pellets are added to the world.
- Power Pellets provide PacMan with the temporary ability to eat the enemies. The enemies turn deep blue, and reverse direction.

This challenge gets you started, but won’t give you all the code.

To help you think this through…

- You will need a new agent (Power Pellet)
- Do you need a new agent or a new shape for the blue ghost?
- When the ghost chases the PacMan, PacMan has a scent of 1000. What happens if he has a scent of -1000? How can you set that new scent?
- How can you limit the time that PacMan’s scent is -1000? Could you create a timer agent that starts counting when it receives a message from PacMan that he ate a Power Pellet? The timer agent should send a message back to PacMan when it is done counting and it’s time for PacMan’s scent to return to 1000.
- Hint: Use the **hill climbing** action rather than all the code for sniffing.

Student Handout 4d:

Challenge 4: Next Level

Next Level

Next Level!

Make a second level for your PacMan!

Before your start this challenge:

You must have a complete basic PacMan game with a PacMan who wins if he eats all the pellets and Ghosts who either move randomly or chase the PacMan. The PacMan loses if a Ghost gets too close. The world should have walls that the Ghost and PacMan cannot cross.

You must have different depictions of the PacMan so that he faces the direction he heads, and he must move continuously.

Description of the Challenge:

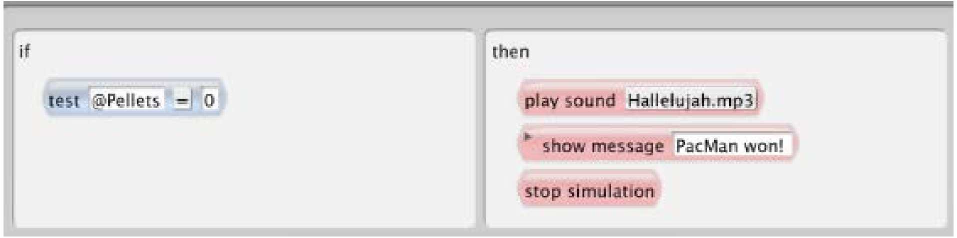
- When the game ends, a new level appears, even harder than before!

This challenge gets you started, but won't give you all the code.

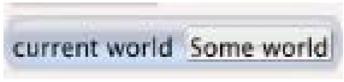
To help you think this through…

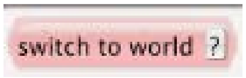
- Do you need a new agent? A new world?
- When would a new level appear?
- What code needs to change to make the new level appear?

You might have a rule like this:



```
if                         then
  test @Pellets = 0          play sound Hallelujah.mp3
                             show message PacMan won!
                             stop simulation
```

How could you use this condition  current world Some world

and this action  switch to world ? to let the player move from a world named "Level 1" to a world named "Level 2"?

Very Important Note: Add another rule that **stops the simulation** if the player has won Level 2 **so the game ends**!