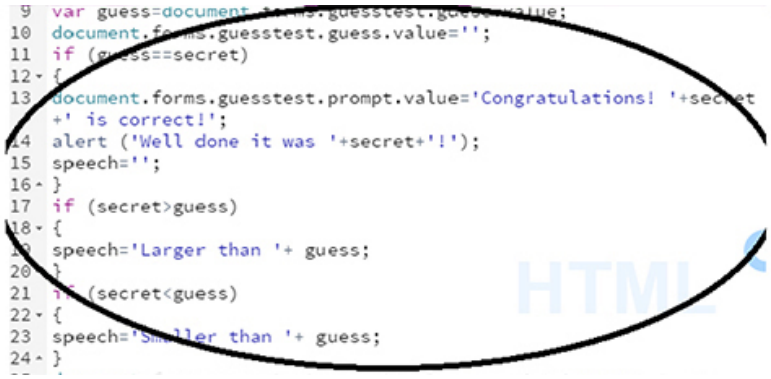
# What does an algorithm look like?

Prompt 2C from the CREATE task:

- Capture and paste the program code segment that implements an algorithm (marked with an oval in section 3 below) that is fundamental for your program to achieve its intended purpose.
- Your code segment must include an algorithm that integrates other algorithms and integrates mathematical and/or logical concepts.
- Describe how each algorithm within your selected algorithm functions independently, as well as in combination with others, to form a new algorithm that helps to achieve the intended purpose of the program.

| | | | |
|---|---|---|---|
| 3: Applying Algorithms<br><br>Submission Requirement: 2c<br><br>LO: 4.1.1; 4.1.2; 5.2.1; 5.5.1<br><br>Weighted: 30% | The selected algorithm is a commonly used algorithm and integrates mathematical and/or logical concepts.<br><br>AND<br>The response provides a general description of the algorithm OR a correct line-by-line summary of the algorithm.<br><br>*If needed, more than one area of the program code can be selected as part of the response to describe the algorithm. | The selected algorithm integrates two or more commonly used or new algorithms and integrates mathematical and/or logical concepts to create a new algorithm.<br><br>AND<br>The response identifies the algorithm's purpose in the program and accurately describes with specificity how the algorithm achieves this purpose.<br><br>*If needed, more than one area of the program code can be selected as part of the response to describe the algorithm. | The selected algorithm integrates two or more commonly used or new algorithms, and integrates mathematical and/or logical concepts to create a new algorithm.<br><br>AND<br>The response identifies the algorithm's purpose in the program and accurately describes with specificity how the algorithm achieves this purpose.<br><br>AND<br>The response accurately describes how two of the algorithms function independently as well as in combination to create a new algorithm.<br><br>*If needed, more than one area of the program code can be selected as part of the response to describe the algorithm. |

| Use Row 3 in the rubric to rate this response. | LOW MEDIUM HIGH |
|---|---|
| When ( Projectile is within 5 meters of Target 1.turret ) becomes true<br>Do: world.TurretHit AI Fire = FireGif 1 AI Turret = Target 1.turret<br><br>When ( Projectile is within 5 meters of Target 2.turret ) becomes true<br>Do: world.TurretHit AI Fire = FireGif 2 AI Turret = Target 2.turret<br><br>When ( Projectile is within 5 meters of Target 3.turret ) becomes true<br>Do: world.TurretHit AI Fire = FireGif 3 AI Turret = Target 3.turret<br><br>When ( Projectile is within 5 meters of Target 4.turret ) becomes true<br>Do: world.TurretHit AI Fire = FireGif 4 AI Turret = Target 4.turret<br><br>When ( Projectile is within 5 meters of Target 5.turret ) becomes true<br>Do: world.TurretHit AI Fire = FireGif 5 AI Turret = Target 5.turret<br><br>Justification: Why did you score this the way you did? Refer to specific sentences in the rubric. | *My code is primarily dependant of its events rather than its methods. These events are what controls when a target tank is hit, and will call the TurretHit method with the specific target tank and its turret as parameters, as well as the fire that appears once a tank is hit. The method Turret Hit simply replaces the targeted tank's turret with a fire animation that will fun for the remainder of the program. Once all of the turrets are hit, a method will be called that will end the game.* |

| Use Row 3 in the rubric to rate this response. | LOW MEDIUM HIGH |
|---|---|
|  <br><br>Justification: Why did you score this the way you did? Refer to specific sentences in the rubric. | *My algorithm is able to implement many others algorithms in multiple ways. It first is able to integrate the variables by creating its own variable for guessing. After this it is able to implement other situations such as if the number is over the secret number or if the number is smaller than the secret number created by the mystery number. This is able to implement previous code created by the mystery number that I had created in my code. Also another portion of code that I created is the use of adding a chat if you win or have not gotten the mystery number. It creates an alert that pops up on your screen if you have achieved success in finding the mystery number as it will say "well done it was 'mystery number'."*<br><br>*This algorithm keeps track of how many of each prize has been collected. In the code just before this segment I create a list that stores the number of each prize that was collected, and set all the values of the elements to 0. I needed to integrate an algorithm for simulating one purchase with an algorithm for continuing until one of each prize is won. The first algorithm is the set block combined with the replace block. I take the current value of a certain element in the list and add 1, to mark one more of that prize.*<br>*The second algorithm involves the repeat block that continues until I have one of each kind of prize. I needed a way to stop this loop when you have a complete set of prizes. The repeat until block has a condition "not countOfEach contains 0." In effect this allows the repeat to continue until the list contains no zeros.* |

| Use Row 3 in the rubric to rate this response. | LOW MEDIUM HIGH |
|---|---|
| ```<br>def fight_beasts():<br>  global step4, step5, planb<br>  step4 = False<br>  step5 = True<br>  planb = True<br>  schrodinger()<br>  buttons(frame)<br><br>def schrodinger():<br>global death,deathpick<br>    deathpick = random.choice(death)<br>    if deathpick == False:<br>        hp_percent(False,25)<br>        text_set("Return with a ", beastpick +" "+<br>random.choice(beast_item)+ ".")<br>    elif deathpick == True and (hp == 0 or hp <0):<br>text_set(you hp_minus(hp)<br>elif true and ! have,died.) (hp deathpick 0 hp ="=">0):<br>        hp_minus(hp)<br>        text_set("You have","died.")<br><br>def buttons(frame):<br>  if step1 == True:<br>     button1 = frame.add_button("Start your journey",<br>startpage, 50)<br>  if step2 == True:<br>     new_2_button(frame,"Take the left path", left_1,"Take the<br>right path",right_1)<br>  if step3 == True and planb == False and left == False:<br>     new_2_button(frame,"Follow a tiny trail.",trail,"Follow a<br>unicorn.",wiz)<br>  if step3 == True and planb == False and left == True:<br>     new_1_button(frame,"Stay for a few days.",camp)<br>  if step3 == True and planb == True and left == True:<br>     new_2_button(frame,"Explore the depths.", entercave,"Find<br>a campsite.",camp)<br>  if step4 == True and planb == False and left == True:<br>     new_1_button(frame,"Continue on.",wiz)<br>  if step4 == True and planb == True and left == True:<br>     new_2_button(frame,"Fight the " + beastpick + ".",<br>fight_beasts,<br>         "Escape in fear.",camp)<br>  if step4 == True and planb == False and left == False:<br>     new_2_button(frame,"Join the fairies in the<br>forest.",fairies,"Return home at your<br>         normal size.",goback)<br>  if step5 == True and planb == True and deathpick == False<br>and left == True:<br>     new_1_button(frame,"Return to the village with your<br>spoils.", goback)<br>  if (step5 == True and planb == True and deathpick == True<br>and left == True and hp == 0)<br>  or hp == 0:<br>     new_1_button(frame,"Start over", restart)<br>  if step5 == True and planb == True and deathpick == True and<br>left == True and hp != 0:<br>     new_1_button(frame,"Escape",camp)<br>  if step6 == True and planb == True and left == False:<br>     new_2_button(frame,"Become the wizard's<br>disciple.",disciple ,"Return with little<br>         knowledge of magic.", goback)<br>  if step7 == True and planb == True and left == False:<br>     new_1_button(frame,"Return to spread magical<br>knowledge.",goback)<br>  if end == True:<br>     no_button(frame)<br>``` | *I chose this algorithm because it is important as it handles what happens when the player in the adventure game fights beasts. When you fight beasts, first you need to change the variables that update which place you are in the game, by using* True *and* False*. Then you have the fight, and then you reset the frames buttons.*<br>*The algorithm* schrodinger() *shown below is integrated into the above algorithm. The algorithm* schrodinger() *is called to determine whether the player lives* (deathpick == False)*, with your health points decreased by 25 percent, or dies and then your health points are set to 0 and a corresponding message appears.*<br><br>*The algorithm buttons(frame) shown below is another integrated algorithm that resets the frame buttons using the buttons() function, which determines how many buttons the game should have next, what they should say, and what function each button should call based on the current state of the game. Both of these algorithms (schrodinger() and buttons(frame)) are noteworthy algorithms that are integrated in the main algorithm in order to make fighting beasts possible in the adventure game.*<br><br>Justification: Why did you score this the way you did? Refer to specific sentences in the rubric. |