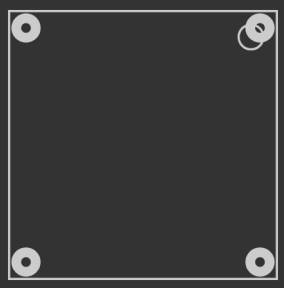
The Octadecayotton (3 Dimensions)



who'll be there when i'm gone?

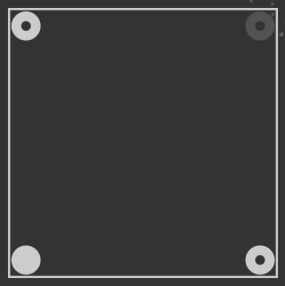
9 dimensions is the default, but each unactivated Octadecayotton reduces the dimension count by 1. Mod settings and missions may also alter the number of dimensions used.

The 8 spheres will start moving, resembling the transformations (rotations and/or reflections) of a 3-dimensional cube. 3 transformations are shown (though mod settings can change this value), followed by a brief pause before starting over.

Identifying Dimensions

- There are 3 dimensions: X, Y, and Z.
- Every sphere represents a corner on the 3-dimensional cube, where 3 lines would meet, all of them being 1 of each axis.
  - For example, a 2-dimensional square has 4 locations where the X and Z lines meet, placing the 4 spheres on those 4 intersections.
- The axes X, Y, and Z represent left/right, toward/away, and up/down respectively. A positive axis means right, toward, or up, and a negative axis means left, away, and down.
- This means that each sphere has 1 of each axis, though each axis is either negative or positive, dictated by their location.
  - Back to the example of a 2-dimensional square, the top-left sphere is -X and +Z, while the bottom-right sphere is +X and -Z.
- Each sphere is color-coded based on how positive it is on the X, Y, and Z plane. With red taking X (right), green taking Y (towards), and blue taking Z (up). A sphere with all axes positive is fully (or as close to) white, and a sphere with all axes negative is fully black.
  - Additive color mixing is used, meaning that high amounts of X and Z, but low amounts of Y will result in approximately magenta.

Identifying Neighbours



- The diagram to the right shows an example of a sphere's 3 neighbours as seen from the front, zoomed in to the bottom-left quarter of the cube.
  - For clarity, the highlighted unfilled spheres are the neighbours of the filled sphere.
- It is important to note that neighbours by definition will be 1 positive/negative away from the original.
  - With a 2-dimensional square, the bottom-left neighbours with the bottom-right, since the only difference is X. The bottom-left is also neighbouring top-left from Z.
- The Y axis cannot be seen from this angle; they point towards the viewer, as do the other axes when viewed from a different angle.
  - This means that every sphere shown in the diagram is actually showing multiple spheres, all stacked on top of each other.
- In reading order (left-to-right then top-to-bottom), the spheres are Z, Y (stacked on top of the filled sphere), and X.
- Because the filled sphere is in a completely negative position (leftmost/backmost/bottommost), going from the filled sphere to any unfilled is travelling to a positive axis, while going from an unfilled sphere to the filled sphere is travelling to a negative axis.

Identifying Transformations

- Look at any 2 spheres that are neighbours and note down the one axis different from each other.
- Determine whether the axis is positive or negative by envisioning an arrow going from one sphere to the other. If the arrow points right/towards/up it's positive, if it's left/away/down it's negative.

Identifying Transformations (...continued)

- After determining their relation to each other, watch them transform.
  - If these 2 spheres are now neighboured from a different axis (rotation) or the same axis but negative (reflection), that implies a transformation.

- Record the new axis that they transformed into, and repeat this process starting with the new axis until the first initial axis is reached.
  - An example would be take axis +X, +X goes to +Z, take axis +Z, +Z goes to -Y, take axis -Y, -Y goes to -X. X has now been reached.

- Every subtransformation has anywhere from 1 to 5 axes involved.

- For any negative initial axis, invert both the initial and new axis. (positive -> negative, negative -> positive)
  - Back to the example, "-Y goes to -X" becomes "+Y goes to +X".

- When the first initial axis has been reached, discard all the initial axes, reverse their order, and combine them.
  - Back to the example, we know that +X goes to +Z, +Z goes to -Y, and +Y goes to +X. Removing all of the initial axes we get +Z, -Y, and +X. Now the order needs to be reversed, getting +X, -Y, and +Z. Finally, combine them, getting +X-Y+Z.

- The resulting sequence is called a subtransformation. In a given transformation there can be 1 to 5 simultaneous subtransformations happening at the same time.

- These subtransformations are completely isolated from each other, as while they do have an effect on the individual sphere's paths, the axes do not, as any axis can only be in 1 subtransformation at any time.

- Subtransformations have multiple ways to be spelled because they are cyclical (they have no beginning or end, and one element affects the next), and subtransformations do not have a concrete order to each other. However, the module does not require either case.
  - +X-Y+Z could be spelled as -Y+Z+X (shift 1 left) and +Z+X-Y (shift 1 right), but not +Z-Y+X (backwards) or +X+Y+Z (inverted negativity), since the order, or positivity/negativity is wrong.

Primary Values

- If there are 0 transformations, skip this page. All primary values are 0.
- All transformations have a primary value. On each subtransformation, create a list of every possible pair of axes from itself and the next axis. (including the last THEN the first axis as a pair*)
  - If the subtransformation is only 1 axis (reflection), ignore the above rule and make only 1 pair of the axis repeated twice.
- There will always be the same number of pairs as there are axes in the subtransformation.
- For each pair, get the value from the table, using the first letter as the row and the second letter as the column.
  - If 1 of 2 axes are negative, multiply the pair's value with -1.

** Even with 2 axes, the rule still applies. Subtransformation +R-T would have pairs +R-T and -T+R.*

	X	Y	Z	
X	1	2	5	X
Y	2	3	6	Y
Z	9	1	4	Z
	X	Y	Z	

Primary Values (...continued)

- For each transformation, take the absolute value of the sum of all pair results from every subtransformation modulo 8. This is the primary value of that transformation. Later in this manual, whenever p_x is used, it refers to the primary value of the x^{th} transformation.

** Modulo is to add/subtract the right number until the left is non-negative, and less than the right.*

The Anchor Sphere

- If there are 0 transformations, skip the next 2 pages. The anchor sphere is located completely negatively. (or black)

- For each transformation, create a_x, where x is the transformation number. All of them start with the value "000".

- Set a_x to be p_x as binary; In other words, subtract the largest number equal or less than p_x found in "Decimal <-> Binary" from it, then set a_x's Xth digit from the left to 1, where X is the position obtained from that same number that was subtracted with. Keep subtracting and setting digits to 1 until p_1 is 0.

Decimal <-> Binary			
Subtract	4	2	1
Position	1st	2nd	3rd

- Look at a_x and its transformation, for each axis, invert the number's position (0 -> 1, 1 -> 0) according to this the table below.

Axis <-> Position			
+Axis	+X	+Y	+Z
-Axis	-Z	-Y	-X
Position	1st	2nd	3rd

The Anchor Sphere (...continued)

- Once all a values are set, create a_0, with the value based on these conditions: (Note that certain a values will not exist if played with lower amounts of rotations than the conditions)
    1. If a_1 exists, and the 2nd digit of a_1 is 1, set the 1st digit of a_0 to 1.
    2. If a_2 exists, and the 2nd digit of a_2 is 1, set the 2nd digit of a_0 to 1.
    3. If a_3 exists, and the 2nd digit of a_3 is 1, set the 3rd digit of a_0 to 1.
- All a-values excluding a_0 is now gray code, convert all gray codes to binary:
    1. The first binary digit will match the first digit of the gray code.
    2. The next digit is a 1 if the sum of the previous digit of the binary code and the current position's gray code is exactly 1. Otherwise, 0.
    3. Repeat step 2 until 3 digits are obtained. This is the binary code.
- XOR a_0 with all other a-values; In other words, add a_0 with a_1, and then refer to the next a. Don't carry ($1+1 \neq 10$) and replace 2's with 0's on each step.
- Replace every 0 with - and every 1 with +.
- This is now the anchor sequence. Whenever the anchor sphere is mentioned, it refers to the only 1 sphere that matches all positive/negative attributes of the anchor sequence's axes.
    - The position of each character represents what axis they belong to, with the order being "XYZ".

Pausing

- Before continuing, note that the module cannot unpause without striking. Make sure all transformations are noted down before proceeding.
- Interact anywhere on the module to pause it. The transformations will stop, and a sound cue is played to indicate that it is ready to be interacted with.
- Each time the module is paused, a random sphere is chosen. This is called the starting sphere. The starting sphere will glow.
- The goal is to get that glowing sphere to be on the same location as the anchor sphere.

Navigation

- When the module is interacted with during the last digit being 0-2, an axis is queued. Each submission from 0-2 represents an axis, though order is random.
- 1 axis need to be queued for a valid input. When the timer's last digit is a 9, it will try submitting the 1 axis. The queue is cleared if any other number of axes are queued.
- The glowing sphere goes to the other side of the 1 axis that were submitted.
- During this submission, all axes can only be submitted up to 2 times.
    - Submitting an axis that has been submitted twice or more times causes a soft-strike. Only the 5^{th} soft-strike causes a strike on the bomb, and resets the module.
    - The module gives no indication of any soft-strikes.
- When the glowing sphere is in the same position as the anchor sphere, submit all 3 axes. The module will strike or solve accordingly.
    - Striking will reset the module.