```
P = {}
(*P simply defines an empty set*)
```

Out[ ]=

```
{}
```

```
b[x_, y_, z_] := Piecewise[{{{{z, 0, 0}, {0, z, 0}, {0, 0, z}}, x == y == 0},
   {{{0, y, y}, {y, 0, y}, {y, y, 0}, {0, y, -y}, {y, 0, -y}, {y, -y, 0}}, x == 0 && y == z},
   {{{0, y, z}, {y, 0, z}, {y, z, 0}, {0, z, y}, {z, 0, y}, {z, y, 0}, {0, -y, z},
     {-y, 0, z}, {-y, z, 0}, {0, z, -y}, {z, 0, -y}, {z, -y, 0}}, x == 0 && y ≠ z},
   {{{x, x, x}, {-x, x, x}, {x, -x, x}, {x, x, -x}}, x == y == z ≠ 0},
   {{{z, x, x}, {x, z, x}, {x, x, z}, {-z, x, x}, {x, -z, x}, {x, x, -z}, {-x, x, z},
     {x, -x, z}, {-x, z, x}, {x, z, -x}, {z, -x, x}, {z, x, -x}}, x == y && x ≠ 0 && z ≠ 0},
   {{{x, y, y}, {y, x, y}, {y, y, x}, {-x, y, y}, {y, -x, y}, {y, y, -x}, {-y, y, x},
     {y, -y, x}, {-y, x, y}, {y, x, -y}, {x, -y, y}, {x, y, -y}}, y == z && y ≠ 0 && x ≠ 0},
   {{{x, y, z}, {-x, y, z}, {x, -y, z}, {x, y, -z}, {y, x, z}, {-y, x, z}, {y, -x, z},
     {y, x, -z}, {y, z, x}, {-y, z, x}, {y, -z, x}, {y, z, -x}, {x, z, y}, {-x, z, y},
     {x, -z, y}, {x, z, -y}, {z, x, y}, {-z, x, y}, {z, -x, y}, {z, x, -y}, {z, y, x},
     {-z, y, x}, {z, -y, x}, {z, y, -x}}, x ≠ y && x ≠ z && y ≠ z && x ≠ 0 && y ≠ 0 && z ≠ 0}}]
(*b is a piecewise function which takes the components of a vector and
 returns a set containing all of the different forms of that vector*)

For[i = 0, i ≤ 15, i = i + 1,
 For[j = i, j ≤ 15, j = j + 1,
  For[k = j, k ≤ 15, k = k + 1,
   If[IntegerExponent[(i^2 + j^2 + k^2), 5] == 0,
    If[GCD[i, j, k] == 1,
     AppendTo[P, b[i, j, k]
     ]
    ]
   ]
  ]
 ]
]
(*This is a loop which checks every integer vector with components smaller
 than 15 to see if its norm squared is divisible by 5. If it is not,
then the loop adds the vector to P, creating a set of vectors
 with norms squared divisible only by primes other than 5*)
```

```
P
(*This is a restatement of P now that we've added all those different elements*)
```

Out[ ]=

```
{{{1, 0, 0}, {0, 1, 0}, {0, 0, 1}},
 {{0, 1, 1}, {1, 0, 1}, {1, 1, 0}, {0, 1, -1}, {1, 0, -1}, {1, -1, 0}},
 ··· 504 ···, {{14, 15, 15}, {15, 14, 15}, {15, 15, 14}, {-14, 15, 15},
  {15, -14, 15}, {15, 15, -14}, {-15, 15, 14}, {15, -15, 14},
  {-15, 14, 15}, {15, 14, -15}, {14, -15, 15}, {14, 15, -15}}}
```

large output | show less | show more | show all | set size limit...

$OutputSizeLimit : This output can only be updated in the same kernel session that generated it.

```
V = Partition[Flatten[P], 3]
(*P is a set of sets,
where each interior set contains all the forms of a certain vector. However,
this makes it difficult for mathematica to run code
 properly. To fix this problem, we create a new list that
 has all the elements of P (this is the flatten command),
but creates new sets, vectors, every three numbers.*)
```

Out[ ]=

```
{{1, 0, 0}, {0, 1, 0}, {0, 0, 1}, {0, 1, 1}, {1, 0, 1}, {1, 1, 0}, {0, 1, -1},
 {1, 0, -1}, {1, -1, 0}, {0, 1, 4}, {1, 0, 4}, {1, 4, 0}, {0, 4, 1}, {4, 0, 1},
 {4, 1, 0}, ··· 9800 ···, {15, -14, 14}, {15, 14, -14}, {14, 15, 15}, {15, 14, 15},
 {15, 15, 14}, {-14, 15, 15}, {15, -14, 15}, {15, 15, -14}, {-15, 15, 14},
 {15, -15, 14}, {-15, 14, 15}, {15, 14, -15}, {14, -15, 15}, {14, 15, -15}}
```

large output | show less | show more | show all | set size limit...

```
W = {{1, 0, 0}, {1, 0, -1}, {1, -1, 0}}
(*This is where we designate what our white vectors will be*)
```

Out[ ]=

```
{{1, 0, 0}, {1, 0, -1}, {1, -1, 0}}
```

```
B = {}
(*This is where we designate our black vectors. I didn't have any so
 I just made this an empty set but one can add them in as necessary.*)
```

Out[ ]=

```
{}
```

```
B' = {}
(*Here we have a new empty list. One can think of this as the set
 for new black vectors that have just been assigned a coloring.*)
Q = Complement[V, B]
(*This is the set of vectors that hasn't been colored black yet,
so we don't continue checking vectors weve already colored.*)
For[i = 1, i ≤ Length[Q], i = i + 1,
 For[k = 1, k ≤ Length[W], k = k + 1,
  If[Q⟦i⟧.W⟦k⟧ == 0,
   AppendTo[B', Q⟦i⟧], Unevaluated[Sequence[]]]]]
(*This loop checks to see if any vectors in Q are
 orthogonal to a white vector. If so, it adds them to B'.*)
MM = DeleteDuplicates[B']
(*This creates a new list containing
 all of the elements from B' without duplicaes.*)
B = Union[B, MM]
(*This adds any new black vectors to B.*)
For[j = 1, j ≤ Length[MM], j = j + 1,
 For[l = j, l ≤ Length[B], l = l + 1,
  If[MM⟦j⟧.B⟦l⟧ == 0,
   AppendTo[W, Cross[MM⟦j⟧, B⟦l⟧] / GCD[Cross[MM⟦j⟧, B⟦l⟧]⟦1⟧,
      Cross[MM⟦j⟧, B⟦l⟧]⟦2⟧, Cross[MM⟦j⟧, B⟦l⟧]⟦3⟧]], Unevaluated[Sequence[]]]]]
(*This loop checks to see if there are any new
 black vectors in MM orthogonal to any vectors in B,
if so it adds the cross product to the list of white vectors *)
For[j = 1, j ≤ Length[MM], j = j + 1,
 For[l = j, l ≤ Length[MM], l = l + 1,
  If[MM⟦j⟧.MM⟦l⟧ == 0,
   AppendTo[W, Cross[MM⟦j⟧, MM⟦l⟧] / GCD[Cross[MM⟦j⟧, MM⟦l⟧]⟦1⟧,
      Cross[MM⟦j⟧, MM⟦l⟧]⟦2⟧, Cross[MM⟦j⟧, MM⟦l⟧]⟦3⟧]], Unevaluated[Sequence[]]]]]
(*This loop checks to see if there are any new black
 vectors in MM orthogonal to any other new black vectors in MM,
if so it adds the cross product to the list of white vectors *)
W = DeleteDuplicates[W]
(*Again, this gets rid of any repetition in the white vectors.*)
```

Out[◦]=

```
{}
```

Out[ ]=

```
{{{1, 0, 0}, {0, 1, 0}, {0, 0, 1}}, ⋯ 660 ⋯ ,
 {{685, 743, 757}, {-685, 743, 757}, {685, -743, 757},
  {685, 743, -757}, {743, 685, 757}, {-743, 685, 757}, {743, -685, 757},
  {743, 685, -757}, {743, 757, 685}, ⋯ 6 ⋯ , {685, 757, -743},
  {757, 685, 743}, {-757, 685, 743}, {757, -685, 743}, {757, 685, -743},
  {757, 743, 685}, {-757, 743, 685}, {757, -743, 685}, {757, 743, -685}}}
```

large output | show less | show more | show all | set size limit...

Out[ ]=

```
{}
```

Out[ ]=

```
{{-1, 1, 1}, {0, 0, 1}, {0, 1, -1}, {0, 1, 0}, {0, 1, 1}, {1, -22, 1}, {1, -5, 1},
 {1, -4, 1}, {1, -2, 1}, {1, -1, 1}, {1, 0, 1}, {1, 1, -22}, {1, 1, -5}, {1, 1, -4},
 {1, 1, -2}, {1, 1, -1}, {1, 1, 0}, {1, 1, 1}, {1, 1, 2}, {1, 1, 4}, {1, 1, 5},
 {1, 1, 22}, {1, 2, 1}, {1, 4, 1}, {1, 5, 1}, {1, 22, 1}, {2, -1, 2}, {2, 1, 2},
 {2, 2, -1}, {2, 2, 1}, {4, -7, 4}, {4, 1, 1}, {4, 4, -7}, {4, 4, 7}, {4, 7, 4},
 {5, -2, 5}, {5, 2, 5}, {5, 5, -2}, {5, 5, 2}, {7, -8, 7}, {7, 7, -8}, {7, 7, 8},
 {7, 8, 7}, {10, -23, 10}, {10, 10, -23}, {10, 10, 23}, {10, 23, 10}, {11, -1, 11},
 {11, 1, 11}, {11, 11, -1}, {11, 11, 1}, {13, -43, 13}, {13, 13, -43}, {13, 13, 43},
 {13, 43, 13}, {17, -112, 17}, {17, 17, -112}, {17, 17, 112}, {17, 112, 17},
 {22, -241, 22}, {22, 22, -241}, {22, 22, 241}, {22, 241, 22}, {23, -20, 23},
 {23, 20, 23}, {23, 23, -20}, {23, 23, 20}, {43, -26, 43}, {43, 26, 43}, {43, 43, -26},
 {43, 43, 26}, {56, -17, 56}, {56, 17, 56}, {56, 56, -17}, {56, 56, 17}, {73, -95, 73},
 {73, 73, -95}, {73, 73, 95}, {73, 95, 73}, {95, -146, 95}, {95, 95, -146},
 {95, 95, 146}, {95, 146, 95}, {197, -526, 197}, {197, 197, -526}, {197, 197, 526},
 {197, 526, 197}, {241, -44, 241}, {241, 44, 241}, {241, 241, -44}, {241, 241, 44},
 {263, -197, 263}, {263, 197, 263}, {263, 263, -197}, {263, 263, 197},
 {329, -920, 329}, {329, 329, -920}, {329, 329, 920}, {329, 920, 329},
 {460, -329, 460}, {460, 329, 460}, {460, 460, -329}, {460, 460, 329}}
```

Out[ ]=

```
{{1, 0, 0}, {1, 0, -1}, {1, -1, 0}, {-1, 0, 0}, {-1, 1, 0},
 {2, -1, -1}, {2, 1, -1}, {-2, 1, -1}, {-1, 0, 1}, {-7, -1, 2},
 {-1, 1, 2}, {-1, 2, 1}, {7, -2, 1}, {-1, 2, 2}, {2, -1, -7}, {-2, 7, 1}}
```

```
Intersection[B, W]
(*checking for a contraditction*)
```

Out[ ]=

```
{}
```

```
B' = {}
Q = Complement[V, B]
For[i = 1, i ≤ Length[Q], i = i + 1,
 For[k = 1, k ≤ Length[W], k = k + 1,
  If[Q〚i〛.W〚k〛 == 0,
   AppendTo[B', Q〚i〛], Unevaluated[Sequence[]]]]]
MM = DeleteDuplicates[B']
B = Union[B, MM]
For[j = 1, j ≤ Length[MM], j = j + 1,
 For[l = j, l ≤ Length[B], l = l + 1,
  If[MM〚j〛.B〚l〛 == 0,
   AppendTo[W, Cross[MM〚j〛, B〚l〛] / GCD[Cross[MM〚j〛, B〚l〛]〚1〛,
      Cross[MM〚j〛, B〚l〛]〚2〛, Cross[MM〚j〛, B〚l〛]〚3〛]], Unevaluated[Sequence[]]]]]
For[j = 1, j ≤ Length[MM], j = j + 1,
 For[l = j, l ≤ Length[MM], l = l + 1,
  If[MM〚j〛.MM〚l〛 == 0,
   AppendTo[W, Cross[MM〚j〛, MM〚l〛] / GCD[Cross[MM〚j〛, MM〚l〛]〚1〛,
      Cross[MM〚j〛, MM〚l〛]〚2〛, Cross[MM〚j〛, MM〚l〛]〚3〛]], Unevaluated[Sequence[]]]]]
W = DeleteDuplicates[W]
(*continuing the process of coloring*)
```

Out[ ]=

```
{}
```

Out[ ]=

```
{{-15, 1, 1}, {-15, 1, 4}, {-15, 1, 5}, {-15, 1, 6}, {-15, 1, 9}, {-15, 1, 10},
 {-15, 1, 11}, {-15, 1, 14}, {-15, 1, 15}, {-15, 2, 2}, {-15, 2, 3},
 {-15, 2, 5}, {-15, 2, 7}, {-15, 2, 8}, … 9032 …, {15, 14, -9}, {15, 14, -6},
 {15, 14, -5}, {15, 14, -4}, {15, 14, -1}, {15, 14, 0}, {15, 14, 1}, {15, 14, 4},
 {15, 14, 5}, {15, 14, 6}, {15, 14, 9}, {15, 14, 10}, {15, 14, 11}, {15, 14, 14}}
```

large output | show less | show more | show all | set si ze limit...

Out[ ]=

```
{{-6, 1, 13}, {-6, 13, 1}, {-5, 1, 11}, {-5, 2, 12}, {-5, 3, 13}, {-5, 4, 14},
 {-5, 11, 1}, {-5, 12, 2}, {-5, 13, 3}, {-5, 14, 4}, {-4, 1, 9}, {-4, 1, 12},
 {-4, 3, 11}, {-4, 6, 13}, {-4, 9, 1}, {-4, 11, 3}, {-4, 12, 1}, {-4, 13, 6},
 {-3, 1, 7}, {-3, 1, 8}, {-3, 1, 13}, {-3, 2, 8}, {-3, 3, 11}, {-3, 7, 1}, {-3, 7, 11},
 {-3, 7, 13}, {-3, 8, 1}, {-3, 8, 2}, {-3, 8, 14}, {-3, 11, 3}, {-3, 11, 7},
 {-3, 11, 13}, {-3, 12, 14}, {-3, 13, 1}, {-3, 13, 7}, {-3, 13, 11}, {-3, 13, 14},
 {-3, 14, 8}, {-3, 14, 12}, {-3, 14, 13}, {-2, 0, 13}, {-2, 2, 9}, {-2, 3, 7},
 {-2, 3, 11}, {-2, 4, 7}, {-2, 4, 13}, {-2, 7, 3}, {-2, 7, 4}, {-2, 7, 9}, {-2, 7, 11},
 {-2, 8, 9}, {-2, 9, 2}, {-2, 9, 7}, {-2, 9, 8}, {-2, 9, 13}, {-2, 11, 3}, {-2, 11, 7},
 {-2, 11, 12}, {-2, 11, 13}, {-2, 12, 11}, {-2, 13, 0}, {-2, 13, 4}, {-2, 13, 9},
 {-2, 13, 11}, {-2, 13, 13}, {-1, 1, 3}, {-1, 1, 9}, {-1, 1, 11}, {-1, 2, 4},
```

```
{-1, 2, 11}, {-1, 3, 1}, {-1, 3, 13}, {-1, 3, 14}, {-1, 4, 2}, {-1, 4, 6}, {-1, 4, 15},
{-1, 6, 4}, {-1, 6, 7}, {-1, 6, 8}, {-1, 7, 6}, {-1, 7, 9}, {-1, 7, 11}, {-1, 8, 6},
{-1, 8, 11}, {-1, 8, 14}, {-1, 9, 1}, {-1, 9, 7}, {-1, 9, 11}, {-1, 11, 1},
{-1, 11, 2}, {-1, 11, 7}, {-1, 11, 8}, {-1, 11, 9}, {-1, 11, 13}, {-1, 12, 14},
{-1, 13, 3}, {-1, 13, 11}, {-1, 14, 3}, {-1, 14, 8}, {-1, 14, 12}, {-1, 15, 4},
{1, -14, 8}, {1, -13, 7}, {1, -12, 14}, {1, -11, 6}, {1, -11, 7}, {1, -11, 8},
{1, -11, 13}, {1, -9, 5}, {1, -9, 7}, {1, -9, 11}, {1, -7, 4}, {1, -7, 6}, {1, -7, 9},
{1, -6, 8}, {1, -6, 13}, {1, -5, 11}, {1, -4, 6}, {1, -4, 9}, {1, -4, 13}, {1, -3, 2},
{1, -3, 7}, {1, -3, 11}, {1, -2, 4}, {1, -2, 9}, {1, -1, 3}, {1, -1, 13}, {1, 2, -3},
{1, 2, 4}, {1, 2, 9}, {1, 3, -1}, {1, 3, 11}, {1, 4, -7}, {1, 4, -2}, {1, 4, 2},
{1, 4, 6}, {1, 4, 9}, {1, 4, 13}, {1, 5, -9}, {1, 5, 6}, {1, 6, -11}, {1, 6, -7},
{1, 6, -4}, {1, 6, 4}, {1, 6, 5}, {1, 6, 8}, {1, 7, -13}, {1, 7, -11}, {1, 7, -9},
{1, 7, -3}, {1, 7, 7}, {1, 7, 9}, {1, 7, 11}, {1, 8, -14}, {1, 8, -11}, {1, 8, -6},
{1, 8, 6}, {1, 8, 9}, {1, 9, -7}, {1, 9, -4}, {1, 9, -2}, {1, 9, 2}, {1, 9, 4},
{1, 9, 7}, {1, 9, 8}, {1, 9, 11}, {1, 11, -9}, {1, 11, -5}, {1, 11, -3}, {1, 11, 3},
{1, 11, 7}, {1, 11, 9}, {1, 11, 13}, {1, 11, 15}, {1, 12, 14}, {1, 13, -11},
{1, 13, -6}, {1, 13, -4}, {1, 13, -1}, {1, 13, 4}, {1, 13, 11}, {1, 14, -12},
{1, 14, 12}, {1, 15, 11}, {2, -13, 4}, {2, -13, 11}, {2, -13, 13}, {2, -13, 14},
{2, -12, 1}, {2, -12, 7}, {2, -12, 11}, {2, -12, 13}, {2, -11, 8}, {2, -11, 12},
{2, -9, 7}, {2, -9, 13}, {2, -8, 3}, {2, -8, 5}, {2, -8, 9}, {2, -7, 6}, {2, -7, 8},
{2, -7, 9}, {2, -7, 11}, {2, -6, 7}, {2, -5, 12}, {2, -4, 3}, {2, -4, 7}, {2, -4, 13},
{2, -3, 1}, {2, -3, 4}, {2, -3, 7}, {2, -3, 8}, {2, -3, 11}, {2, -2, 3}, {2, -2, 9},
{2, -1, 3}, {2, -1, 4}, {2, -1, 8}, {2, -1, 12}, {2, 0, 7}, {2, 1, -12}, {2, 1, -3},
{2, 1, 3}, {2, 1, 8}, {2, 1, 12}, {2, 3, -8}, {2, 3, -4}, {2, 3, -2}, {2, 3, -1},
{2, 3, 1}, {2, 3, 7}, {2, 3, 11}, {2, 4, -13}, {2, 4, -3}, {2, 4, -1}, {2, 4, 9},
{2, 5, -8}, {2, 6, -7}, {2, 6, 7}, {2, 7, -12}, {2, 7, -9}, {2, 7, -6}, {2, 7, -4},
{2, 7, -3}, {2, 7, 0}, {2, 7, 3}, {2, 7, 6}, {2, 7, 8}, {2, 7, 9}, {2, 7, 11},
{2, 7, 14}, {2, 8, -11}, {2, 8, -7}, {2, 8, -3}, {2, 8, -1}, {2, 8, 1}, {2, 8, 7},
{2, 8, 11}, {2, 9, -8}, {2, 9, -7}, {2, 9, -2}, {2, 9, 4}, {2, 9, 7}, {2, 9, 13},
{2, 9, 14}, {2, 11, -13}, {2, 11, -12}, {2, 11, -7}, {2, 11, -3}, {2, 11, 3},
{2, 11, 7}, {2, 11, 8}, {2, 12, -11}, {2, 12, -5}, {2, 12, -1}, {2, 12, 1},
{2, 12, 13}, {2, 13, -13}, {2, 13, -12}, {2, 13, -9}, {2, 13, -4}, {2, 13, 9},
{2, 13, 12}, {2, 14, -13}, {2, 14, 7}, {2, 14, 9}, {3, -14, 8}, {3, -13, 2},
{3, -13, 4}, {3, -13, 8}, {3, -13, 13}, {3, -13, 14}, {3, -12, 11}, {3, -12, 14},
{3, -11, 7}, {3, -11, 13}, {3, -9, 13}, {3, -8, 14}, {3, -7, 5}, {3, -7, 7},
{3, -7, 9}, {3, -7, 11}, {3, -7, 13}, {3, -5, 8}, {3, -5, 13}, {3, -4, 2},
{3, -4, 7}, {3, -4, 11}, {3, -3, 11}, {3, -2, 4}, {3, -2, 7}, {3, -2, 8}, {3, -1, 2},
{3, -1, 7}, {3, -1, 8}, {3, -1, 13}, {3, 1, 1}, {3, 1, 7}, {3, 1, 11}, {3, 1, 13},
{3, 2, -13}, {3, 2, -4}, {3, 2, -1}, {3, 2, 4}, {3, 2, 8}, {3, 4, -13}, {3, 4, -2},
{3, 4, 2}, {3, 4, 7}, {3, 5, -7}, {3, 5, 13}, {3, 7, -11}, {3, 7, -7}, {3, 7, -4},
{3, 7, -2}, {3, 7, -1}, {3, 7, 1}, {3, 7, 4}, {3, 7, 7}, {3, 7, 9}, {3, 7, 11},
{3, 7, 13}, {3, 7, 14}, {3, 8, -14}, {3, 8, -13}, {3, 8, -5}, {3, 8, -2}, {3, 8, -1},
{3, 8, 2}, {3, 8, 11}, {3, 8, 14}, {3, 9, -7}, {3, 9, 7}, {3, 11, -12}, {3, 11, -7},
{3, 11, -4}, {3, 11, -3}, {3, 11, 1}, {3, 11, 7}, {3, 11, 8}, {3, 13, -13},
```

```
{3, 13, -11}, {3, 13, -9}, {3, 13, -7}, {3, 13, -5}, {3, 13, -1}, {3, 13, 1},
{3, 13, 5}, {3, 13, 7}, {3, 13, 13}, {3, 14, -13}, {3, 14, -12}, {3, 14, -8},
{3, 14, 7}, {3, 14, 8}, {4, -14, 7}, {4, -14, 9}, {4, -13, 1}, {4, -12, 11},
{4, -11, 5}, {4, -11, 6}, {4, -11, 7}, {4, -11, 13}, {4, -10, 9}, {4, -9, 2},
{4, -9, 11}, {4, -8, 1}, {4, -8, 9}, {4, -7, 1}, {4, -7, 6}, {4, -7, 9}, {4, -6, 5},
{4, -6, 11}, {4, -6, 13}, {4, -5, 14}, {4, -4, 7}, {4, -3, 11}, {4, -3, 14},
{4, -2, 3}, {4, -2, 13}, {4, -1, 3}, {4, -1, 6}, {4, -1, 7}, {4, -1, 9}, {4, -1, 10},
{4, -1, 12}, {4, 1, -13}, {4, 1, -8}, {4, 1, -7}, {4, 1, 1}, {4, 1, 2}, {4, 1, 7},
{4, 1, 8}, {4, 1, 9}, {4, 1, 13}, {4, 2, -9}, {4, 2, 1}, {4, 2, 9}, {4, 3, -2},
{4, 3, -1}, {4, 3, 11}, {4, 3, 14}, {4, 5, -11}, {4, 5, -6}, {4, 6, -11}, {4, 6, -7},
{4, 6, -1}, {4, 6, 7}, {4, 7, -14}, {4, 7, -11}, {4, 7, -4}, {4, 7, -1}, {4, 7, 1},
{4, 7, 6}, {4, 7, 11}, {4, 8, 1}, {4, 8, 9}, {4, 9, -14}, {4, 9, -10}, {4, 9, -8},
{4, 9, -7}, {4, 9, -1}, {4, 9, 1}, {4, 9, 2}, {4, 9, 8}, {4, 9, 14}, {4, 10, -1},
{4, 11, -12}, {4, 11, -9}, {4, 11, -6}, {4, 11, -3}, {4, 11, 3}, {4, 11, 7},
{4, 11, 12}, {4, 12, -1}, {4, 12, 11}, {4, 13, -11}, {4, 13, -6}, {4, 13, -2},
{4, 13, 1}, {4, 13, 13}, {4, 14, -5}, {4, 14, -3}, {4, 14, 3}, {4, 14, 9}, {5, -15, 7},
{5, -11, 6}, {5, -10, 9}, {5, -9, 1}, {5, -8, 2}, {5, -7, 3}, {5, -7, 7}, {5, -6, 4},
{5, -5, 11}, {5, -4, 6}, {5, -4, 14}, {5, -3, 7}, {5, -3, 8}, {5, -3, 13}, {5, -2, 8},
{5, -2, 12}, {5, -1, 9}, {5, -1, 11}, {5, 0, 13}, {5, 1, -9}, {5, 1, 9}, {5, 1, 11},
{5, 2, -8}, {5, 2, 8}, {5, 2, 12}, {5, 3, -7}, {5, 3, 7}, {5, 3, 13}, {5, 4, -6},
{5, 4, 6}, {5, 4, 14}, {5, 6, -11}, {5, 6, -4}, {5, 6, 4}, {5, 7, -15}, {5, 7, -7},
{5, 7, -3}, {5, 7, 3}, {5, 8, -3}, {5, 8, -2}, {5, 8, 2}, {5, 9, -10}, {5, 9, -1},
{5, 9, 1}, {5, 9, 11}, {5, 11, -5}, {5, 11, -1}, {5, 11, 1}, {5, 11, 9}, {5, 12, -2},
{5, 12, 2}, {5, 12, 13}, {5, 13, -3}, {5, 13, 0}, {5, 13, 3}, {5, 13, 12}, {5, 14, -4},
{5, 14, 4}, {6, -14, 7}, {6, -13, 1}, {6, -13, 4}, {6, -13, 11}, {6, -13, 14},
{6, -11, 1}, {6, -11, 3}, {6, -11, 8}, {6, -11, 13}, {6, -11, 14}, {6, -9, 10},
{6, -8, 7}, {6, -8, 11}, {6, -7, 4}, {6, -7, 9}, {6, -7, 11}, {6, -6, 11}, {6, -4, 5},
{6, -4, 7}, {6, -1, 4}, {6, -1, 8}, {6, -1, 11}, {6, -1, 13}, {6, 1, -13}, {6, 1, -11},
{6, 1, 2}, {6, 1, 4}, {6, 1, 8}, {6, 1, 11}, {6, 1, 13}, {6, 2, 1}, {6, 2, 11},
{6, 3, -11}, {6, 3, 11}, {6, 3, 14}, {6, 4, -13}, {6, 4, -7}, {6, 4, -1}, {6, 4, 1},
{6, 4, 7}, {6, 4, 13}, {6, 5, -4}, {6, 7, -14}, {6, 7, -8}, {6, 7, -4}, {6, 7, 4},
{6, 7, 9}, {6, 7, 11}, {6, 8, -11}, {6, 8, -1}, {6, 8, 1}, {6, 8, 11}, {6, 9, -7},
{6, 9, 7}, {6, 10, -9}, {6, 11, -13}, {6, 11, -8}, {6, 11, -7}, {6, 11, -6},
{6, 11, -1}, {6, 11, 1}, {6, 11, 2}, {6, 11, 3}, {6, 11, 7}, {6, 11, 8}, {6, 11, 13},
{6, 13, -11}, {6, 13, -1}, {6, 13, 1}, {6, 13, 4}, {6, 13, 11}, {6, 13, 14},
{6, 14, -13}, {6, 14, -11}, {6, 14, 3}, {6, 14, 13}, {7, -13, 1}, {7, -13, 9},
{7, -13, 10}, {7, -13, 13}, {7, -12, 2}, {7, -12, 11}, {7, -11, 3}, {7, -11, 9},
{7, -11, 13}, {7, -9, 8}, {7, -8, 6}, {7, -8, 15}, {7, -6, 8}, {7, -3, 5}, {7, -3, 11},
{7, -3, 13}, {7, -2, 11}, {7, -2, 12}, {7, -1, 4}, {7, -1, 9}, {7, -1, 12},
{7, -1, 13}, {7, 1, -13}, {7, 1, 3}, {7, 1, 13}, {7, 2, -12}, {7, 2, 3}, {7, 2, 12},
{7, 3, -11}, {7, 3, 1}, {7, 3, 2}, {7, 3, 11}, {7, 3, 13}, {7, 4, -1}, {7, 5, -3},
{7, 6, -8}, {7, 6, 8}, {7, 8, -9}, {7, 8, -6}, {7, 8, 6}, {7, 9, -13}, {7, 9, -11},
{7, 9, -1}, {7, 10, -13}, {7, 11, -12}, {7, 11, -3}, {7, 11, -2}, {7, 11, 3},
{7, 11, 12}, {7, 11, 13}, {7, 12, -2}, {7, 12, -1}, {7, 12, 2}, {7, 12, 11},
```

```
{7, 13, -13}, {7, 13, -11}, {7, 13, -3}, {7, 13, -1}, {7, 13, 1}, {7, 13, 3},
{7, 13, 11}, {7, 15, -8}, {8, -14, 3}, {8, -14, 11}, {8, -13, 3}, {8, -13, 9},
{8, -13, 13}, {8, -13, 14}, {8, -12, 11}, {8, -9, 7}, {8, -9, 13}, {8, -7, 7},
{8, -7, 9}, {8, -7, 11}, {8, -7, 15}, {8, -6, 7}, {8, -6, 13}, {8, -4, 13},
{8, -3, 7}, {8, -3, 13}, {8, -3, 14}, {8, -2, 5}, {8, -2, 9}, {8, 1, 3}, {8, 1, 6},
{8, 1, 11}, {8, 2, 3}, {8, 2, 9}, {8, 3, -14}, {8, 3, -13}, {8, 3, 1}, {8, 3, 2},
{8, 3, 5}, {8, 3, 13}, {8, 3, 14}, {8, 5, -2}, {8, 5, 3}, {8, 6, 1}, {8, 6, 7},
{8, 6, 13}, {8, 7, -9}, {8, 7, -7}, {8, 7, -6}, {8, 7, -3}, {8, 7, 6}, {8, 7, 9},
{8, 9, -13}, {8, 9, -7}, {8, 9, -2}, {8, 9, 2}, {8, 9, 7}, {8, 11, -14}, {8, 11, -12},
{8, 11, -7}, {8, 11, 1}, {8, 13, -13}, {8, 13, -9}, {8, 13, -6}, {8, 13, -4},
{8, 13, -3}, {8, 13, 3}, {8, 13, 6}, {8, 13, 14}, {8, 13, 15}, {8, 14, -13},
{8, 14, -3}, {8, 14, 3}, {8, 14, 13}, {8, 15, -7}, {8, 15, 13}, {9, -14, 4},
{9, -13, 11}, {9, -12, 14}, {9, -11, 7}, {9, -11, 10}, {9, -11, 13}, {9, -9, 7},
{9, -7, 8}, {9, -7, 11}, {9, -7, 14}, {9, -4, 14}, {9, -3, 11}, {9, -2, 13},
{9, -1, 5}, {9, -1, 11}, {9, 1, 4}, {9, 1, 7}, {9, 3, 11}, {9, 4, -14}, {9, 4, 1},
{9, 4, 14}, {9, 5, -1}, {9, 7, -11}, {9, 7, -9}, {9, 7, 1}, {9, 7, 11}, {9, 8, -7},
{9, 10, -11}, {9, 11, -13}, {9, 11, -7}, {9, 11, -3}, {9, 11, -1}, {9, 11, 3},
{9, 11, 7}, {9, 12, 14}, {9, 13, -11}, {9, 13, -2}, {9, 14, -12}, {9, 14, -7},
{9, 14, -4}, {9, 14, 4}, {9, 14, 12}, {10, -13, 7}, {10, -13, 13}, {10, -11, 9},
{10, -10, 9}, {10, -9, 11}, {10, -9, 14}, {10, -8, 13}, {10, -7, 12}, {10, -7, 13},
{10, -6, 11}, {10, -5, 8}, {10, -4, 9}, {10, -3, 8}, {10, -2, 7}, {10, -1, 6},
{10, 0, 7}, {10, 1, 4}, {10, 2, 3}, {10, 3, 2}, {10, 4, 1}, {10, 4, 15}, {10, 5, 6},
{10, 6, -1}, {10, 6, 5}, {10, 7, -13}, {10, 7, -2}, {10, 7, 0}, {10, 7, 13},
{10, 8, -5}, {10, 8, -3}, {10, 9, -11}, {10, 9, -10}, {10, 9, -4}, {10, 9, 11},
{10, 11, -9}, {10, 11, -6}, {10, 11, 9}, {10, 12, -7}, {10, 13, -13}, {10, 13, -8},
{10, 13, -7}, {10, 13, 7}, {10, 13, 13}, {10, 14, -9}, {10, 15, 4}, {11, -14, 8},
{11, -13, 9}, {11, -13, 12}, {11, -9, 10}, {11, -9, 13}, {11, -8, 14}, {11, -7, 9},
{11, -3, 7}, {11, -1, 6}, {11, -1, 13}, {11, 1, 5}, {11, 1, 9}, {11, 2, 7}, {11, 3, 4},
{11, 4, 3}, {11, 5, 1}, {11, 6, -1}, {11, 7, -3}, {11, 7, 2}, {11, 8, -14},
{11, 8, 14}, {11, 9, -13}, {11, 9, -7}, {11, 9, 1}, {11, 9, 13}, {11, 10, -9},
{11, 12, -13}, {11, 13, -9}, {11, 13, -1}, {11, 13, 9}, {11, 14, -8}, {11, 14, 8},
{12, -14, 13}, {12, -13, 11}, {12, -12, 11}, {12, -11, 13}, {12, -8, 11},
{12, -7, 11}, {12, -7, 13}, {12, -4, 13}, {12, -3, 8}, {12, -2, 7}, {12, -1, 7},
{12, -1, 13}, {12, -1, 14}, {12, 0, 7}, {12, 1, 13}, {12, 2, 5}, {12, 2, 15},
{12, 4, 9}, {12, 4, 13}, {12, 5, 2}, {12, 7, -2}, {12, 7, -1}, {12, 7, 0}, {12, 7, 7},
{12, 7, 11}, {12, 8, -3}, {12, 8, 11}, {12, 9, 4}, {12, 11, -13}, {12, 11, -12},
{12, 11, -8}, {12, 11, -7}, {12, 11, 7}, {12, 11, 8}, {12, 11, 13}, {12, 13, -14},
{12, 13, -11}, {12, 13, -7}, {12, 13, -4}, {12, 13, -1}, {12, 13, 1}, {12, 13, 4},
{12, 13, 11}, {12, 14, -1}, {12, 15, 2}, {13, -14, 12}, {13, -12, 14}, {13, -11, 12},
{13, -9, 11}, {13, -7, 10}, {13, -3, 8}, {13, -1, 7}, {13, 1, 6}, {13, 1, 11},
{13, 2, 9}, {13, 3, 5}, {13, 3, 7}, {13, 5, 3}, {13, 6, 1}, {13, 7, -1}, {13, 7, 3},
{13, 8, -3}, {13, 9, 2}, {13, 10, -7}, {13, 11, -9}, {13, 11, 1}, {13, 12, -14},
{13, 12, -11}, {13, 12, 14}, {13, 14, -12}, {13, 14, 12}, {14, -13, 11},
{14, -12, 13}, {14, -11, 12}, {14, -8, 11}, {14, -6, 11}, {14, -6, 13}, {14, -4, 9},
```

```
{14, -4, 11}, {14, -2, 9}, {14, -1, 8}, {14, -1, 10}, {14, 1, 6}, {14, 1, 12},
{14, 3, 4}, {14, 3, 8}, {14, 4, 3}, {14, 4, 5}, {14, 4, 9}, {14, 5, 4}, {14, 6, 1},
{14, 8, -1}, {14, 8, 3}, {14, 8, 9}, {14, 9, -4}, {14, 9, -2}, {14, 9, 4},
{14, 9, 8}, {14, 10, -1}, {14, 11, -13}, {14, 11, -8}, {14, 11, -6}, {14, 11, -4},
{14, 11, 12}, {14, 11, 13}, {14, 12, -11}, {14, 12, 1}, {14, 12, 11}, {14, 13, -12},
{14, 13, -6}, {14, 13, 11}, {15, -13, 13}, {15, -6, 11}, {15, -5, 11}, {15, -1, 10},
{15, 0, 7}, {15, 1, 9}, {15, 3, 5}, {15, 5, 3}, {15, 7, 0}, {15, 7, 8}, {15, 8, 7},
{15, 9, 1}, {15, 10, -1}, {15, 11, -6}, {15, 11, -5}, {15, 13, -13}, {15, 13, 13}}
```

Out[ ]=

```
{{-6, 1, 13}, {-6, 13, 1}, {-5, 1, 11}, {-5, 2, 12}, {-5, 3, 13}, {-5, 4, 14},
 {-5, 11, 1}, {-5, 12, 2}, {-5, 13, 3}, {-5, 14, 4}, {-4, 1, 9}, {-4, 1, 12},
 {-4, 3, 11}, {-4, 6, 13}, {-4, 9, 1}, … 1716 …, {15, 15, -11}, {15, 15, -8},
 {15, 15, -7}, {15, 15, -4}, {15, 15, -2}, {15, 15, -1}, {15, 15, 1}, {15, 15, 2},
 {15, 15, 4}, {15, 15, 7}, {15, 15, 8}, {15, 15, 11}, {15, 15, 13}, {15, 15, 14}}
```

large output | show less | show more | show all | set size limit...

Out[ ]=

```
{{1, 0, 0}, {1, 0, -1}, {1, -1, 0}, {-1, 0, 0}, {-421, 210, 196},
 {-421, 225, 210}, {-394, 195, 169}, {-394, 225, 195},
 {-313, 156, 144}, {-313, 169, 156}, {-317, 154, 121}, … 3023 …,
 {121, -260, -281}, {-143, 199, 364}, {-67, 256, 231}, {-71, 334, 279},
 {-71, 271, 220}, {-49, 274, 105}, {49, -105, -274}, {67, -231, -256},
 {71, -279, -334}, {71, -220, -271}, {143, -364, -199}}
```

large output | show less | show more | show all | set size limit...

```
Intersection[B, W]
(*checking for a contradiction, and we get one!*)
```

Out[ ]=

```
{{-3, 8, 2}, {2, 2, -5}, {4, 2, 1}}
```

```
(*In case we don't get a coloring right off the bat,
we continue coloring until we have a step where no new black vectors are added,
then we run the code one more time to make sure there are no new
 white vectors which we haven't checked yet, and then we're done. The
 code has colored all it can from the black and white vectors given*)
```