Vector_2D_Description

May 25, 2024

[ ]:
```python
import math


class Vector:
    """2D Vector Class"""

    def __init__(self, x=0, y=0):
        self.x = x
        self.y = y

    def __repr__(self) -> str:
        return f"Vector({self.x}, {self.y})"

    def __str__(self) -> str:
        return f"({self.x},{self.y})"

    def __eq__(self, other) -> bool:
        return self.x == other.x and self.y == other.y

    def __add__(self, other) -> "Vector":
        """add"""
        return Vector(self.x + other.x, self.y + other.y)

    def __sub__(self, other) -> "Vector":
        """substract"""
        return Vector(self.x - other.x, self.y - other.y)

    def __mul__(self, lamda: float) -> "Vector":
        """multiply"""
        return Vector(self.x * lamda, self.y * lamda)

    def __abs__(self) -> float:
        """length"""
        return (self.x**2 + self.y**2) ** 0.5

    def dot(self, other) -> float:
        """dot product"""
        return self.x * other.x + self.y * other.y

    def normalize(self) -> "Vector":
        """normalize"""
        return Vector(self.x / abs(self), self.y / abs(self))

    def angle(self, other) -> float:
        """angle(without orientation)"""
        theta = math.acos(self.dot(other) / (abs(self) * abs(other)))
        return round(math.degrees(theta), 2)

    def project(self, other) -> "Vector":
        """projection"""
        return other.normalize() * self.dot(other.normalize())

    def rotate(self, angle_deg: float) -> "Vector":
        """rotate anticlockwise"""
        angle_rad = math.radians(angle_deg)
        new_x = self.x * math.cos(angle_rad) - self.y * math.sin(angle_rad)
        new_x = round(new_x, 2)
        new_y = self.x * math.sin(angle_rad) + self.y * math.cos(angle_rad)
        new_y = round(new_y, 2)
        return Vector(new_x, new_y)

    def is_perpendicular(self, other):
        return abs(self.dot(other)) < 1e-10

    def reflect(self, other):
        """reflection"""
        return self.project(other) * 2 - self
```

[ ]:
```
help(Vector)
```

```
Help on class Vector in module __main__:

class Vector(builtins.object)
 |  Vector(x=0, y=0)
 |
 |  2D Vector Class
 |
 |  Methods defined here:
 |
 |  __abs__(self) -> float
 |      length
 |
 |  __add__(self, other) -> 'Vector'
 |      add
 |
 |  __eq__(self, other) -> bool
 |      Return self==value.
 |
 |  __init__(self, x=0, y=0)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __mul__(self, lamda: float) -> 'Vector'
 |      multiply
 |
 |  __repr__(self) -> str
 |      Return repr(self).
 |
 |  __str__(self) -> str
 |      Return str(self).
 |
 |  __sub__(self, other) -> 'Vector'
 |      substract
 |
 |  angle(self, other) -> float
 |      angle(without orientation)
 |
 |  dot(self, other) -> float
 |      dot product
 |
 |  is_perpendicular(self, other)
 |
 |  normalize(self) -> 'Vector'
 |      normalize
 |
 |  project(self, other) -> 'Vector'
 |      projection
 |
 |  reflect(self, other)
 |      reflection
 |
 |  rotate(self, angle_deg: float) -> 'Vector'
 |      rotate anticlockwise
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None
```