Project Documentation: Calculator 2D

Katarzyna Trzos

June 2, 2024

1 Classes

This document provides an overview of the classes implemented in the project: `Vector`, `Point`, `Segment`, `Line`, `Parable`, `Triangle`, `Square`, `Rectangle`, and `Circle`.

1.1 Class Vector

Attributes:

- `double x;` - The x-coordinate of the vector.
- `double y;` - The y-coordinate of the vector.

Methods:

- `Vector();` - Default constructor.
- `Vector(const Vector &v);` - Copy constructor.
- `Vector(double x, double y);` - Constructor that initializes the vector with given x and y coordinates.
- `Vector(const Point &A, const Point &B);` - Constructor that creates a vector from two points A and B.
- `Vector(const Segment &AB);` - Constructor that creates a vector from a segment.
- `void set_vector(double x, double y);` - Sets the x and y coordinates of the vector.
- `double getx() const;` - Returns the x-coordinate of the vector.
- `double gety() const;` - Returns the y-coordinate of the vector.
- `double length() const;` - Returns the length (magnitude) of the vector.
- `friend bool operator==(const Vector &v, const Vector &u);` - Checks if two vectors are equal.

- `friend bool operator!=(const Vector &v, const Vector &u);` - Checks if two vectors are not equal.
- `friend Vector operator+(const Vector &v, const Vector &u);` - Adds two vectors.
- `friend Vector operator-(const Vector &v, const Vector &u);` - Subtracts vector u from vector v.
- `friend Vector operator*(const Vector &v, double c);` - Multiplies vector v by scalar c.
- `friend double operator*(const Vector &v, const Vector &u);` - Calculates the dot product of two vectors.
- `Vector &operator+=(const Vector &v);` - Adds vector v to the current vector.
- `Vector &operator-=(const Vector &v);` - Subtracts vector v from the current vector.
- `Vector operator*=(double c);` - Multiplies the current vector by scalar c.

1.2 Class Point

Attributes:

- `double x;` - The x-coordinate of the point.
- `double y;` - The y-coordinate of the point.

Methods:

- `Point();` - Default constructor.
- `Point(const Point &p);` - Copy constructor.
- `Point(double x, double y);` - Constructor that initializes the point with given x and y coordinates.
- `void set_point(double x, double y);` - Sets the x and y coordinates of the point.
- `double getx() const;` - Returns the x-coordinate of the point.
- `double gety() const;` - Returns the y-coordinate of the point.
- `void translate(const Vector &v);` - Translates the point by the given vector.
- `void symmetry(const Line &k);` - Reflects the point across the given line.

- `friend bool operator==(const Point &A, const Point &B);` - Checks if two points are equal.
- `friend bool operator!=(const Point &A, const Point &B);` - Checks if two points are not equal.

1.3 Class Segment

Attributes:

- `Point A;` - The start point of the segment.
- `Point B;` - The end point of the segment.

Methods:

- `Segment();` - Default constructor.
- `Segment(const Segment &s);` - Copy constructor.
- `Segment(const Point &A, const Point &B);` - Constructor that initializes the segment with points A and B.
- `Segment(const Vector &v);` - Constructor that creates a segment from a vector.
- `void set_segment(const Point &A, const Point &B);` - Sets the points A and B of the segment.
- `Point getA() const;` - Returns the start point of the segment.
- `Point getB() const;` - Returns the end point of the segment.
- `void translate(const Vector &v);` - Translates the segment by the given vector.
- `void symmetry(const Line &k);` - Reflects the segment across the given line.
- `double length() const;` - Returns the length of the segment.
- `bool is_parallel(const Segment &AB);` - Checks if the segment is parallel to another segment AB.
- `bool is_perpendicular(const Segment &AB);` - Checks if the segment is perpendicular to another segment AB.
- `bool belong(const Point &A);` - Checks if point A belongs to the segment.
- `bool intersect(const Segment &AB);` - Checks if the segment intersects with another segment AB.

- `friend bool operator==(const Segment &AB, const Segment &CD);` - Checks if two segments are equal.
- `friend bool operator!=(const Segment &AB, const Segment &CD);` - Checks if two segments are not equal.

1.4 Class Line

Attributes:

- `double a;` - The slope of the line.
- `double b;` - The y-intercept of the line.

Methods:

- `Line();` - Default constructor.
- `Line(const Line &s);` - Copy constructor.
- `Line(const Point &A, const Point &B);` - Constructor that initializes the line with points A and B.
- `Line(double a, double b);` - Constructor that initializes the line with slope a and y-intercept b.
- `Line(const Vector &v);` - Constructor that creates a line from a vector.
- `Line(const Segment &AB);` - Constructor that creates a line from a segment.
- `void set_Line(double da, double db);` - Sets the slope and y-intercept of the line.
- `double geta() const;` - Returns the slope of the line.
- `double getb() const;` - Returns the y-intercept of the line.
- `void translate(const Vector &v);` - Translates the line by the given vector.
- `void symmetry(const Line &k);` - Reflects the line across the given line.
- `double value_at(double x);` - Returns the y value at a given x value on the line.
- `double zero_of_fun(double x);` - Returns the x value at which the line crosses the x-axis.
- `bool belong(const Point &x);` - Checks if a point belongs to the line.
- `bool is_parallel(const Line &k);` - Checks if the line is parallel to another line k.

- `bool is_perpendicular(const Line &k);` - Checks if the line is perpendicular to another line k.
- `Line perpendicular_at(const Point &A);` - Returns the line perpendicular to the current line passing through point A.
- `friend bool operator==(const Line &k, const Line &l);` - Checks if two lines are equal.
- `friend bool operator!=(const Line &k, const Line &l);` - Checks if two lines are not equal.

1.5 Class Triangle

Attributes:

- `Point A;` - The first vertex of the triangle.
- `Point B;` - The second vertex of the triangle.
- `Point C;` - The third vertex of the triangle.

Methods:

- `Triangle();` - Default constructor.
- `Triangle(const Triangle &t);` - Copy constructor.
- `Triangle(Point A, Point B, Point C);` - Constructor that initializes the triangle with vertices A, B, and C.
- `void set_triangle(Point A, Point B, Point C);` - Sets the vertices A, B, and C of the triangle.
- `Point getA() const;` - Returns the first vertex of the triangle.
- `Point getB() const;` - Returns the second vertex of the triangle.
- `Point getC() const;` - Returns the third vertex of the triangle.
- `void translate(const Vector &v);` - Translates the triangle by the given vector.
- `void symmetry(const Line &k);` - Reflects the triangle across the given line.
- `double perimeter() const;` - Returns the perimeter of the triangle.
- `double area() const;` - Returns the area of the triangle.
- `bool is_inside(const Point &d);` - Checks if point d is inside the triangle.

- `bool isRightAngled() const;` - Checks if the triangle is right-angled.
- `bool isAcuteAngled() const;` - Checks if the triangle is acute-angled.
- `bool isObtuseAngled() const;` - Checks if the triangle is obtuse-angled.
- `Circle Incircle() const;` - Returns the incircle of the triangle.
- `Circle Circumcircle() const;` - Returns the circumcircle of the triangle.
- `friend bool operator==(const Triangle &t1, const Triangle &t2);` - Checks if two triangles are equal.
- `friend bool operator!=(const Triangle &t1, const Triangle &t2);` - Checks if two triangles are not equal.

1.6 Class Square

Attributes:

- `Point A;` - The first vertex of the square.
- `Point B;` - The second vertex of the square.
- `Point C;` - The third vertex of the square.
- `Point D;` - The fourth vertex of the square.

Methods:

- `Square();` - Default constructor.
- `Square(const Square &s);` - Copy constructor.
- `Square(const Point &A, const Point &B, const Point &C, const Point &D);` - Constructor that initializes the square with vertices A, B, C, and D.
- `void set_square(const Point &A, const Point &B, const Point &C, const Point &D);` - Sets the vertices A, B, C, and D of the square.
- `Point getA() const;` - Returns the first vertex of the square.
- `Point getB() const;` - Returns the second vertex of the square.
- `Point getC() const;` - Returns the third vertex of the square.
- `Point getD() const;` - Returns the fourth vertex of the square.
- `void translate(const Vector &v);` - Translates the square by the given vector.

- `void symmetry(const Line &k);` - Reflects the square across the given line.
- `double perimeter() const;` - Returns the perimeter of the square.
- `double area() const;` - Returns the area of the square.
- `bool inside(const Point &A);` - Checks if point A is inside the square.
- `friend bool operator==(const Square &s1, const Square &s2);` - Checks if two squares are equal.
- `friend bool operator!=(const Square &s1, const Square &s2);` - Checks if two squares are not equal.

1.7 Class Rectangle

Attributes:

- `Point A;` - The first vertex of the rectangle.
- `Point B;` - The second vertex of the rectangle.
- `Point C;` - The third vertex of the rectangle.
- `Point D;` - The fourth vertex of the rectangle.

Methods:

- `Rectangle();` - Default constructor.
- `Rectangle(const Rectangle &t);` - Copy constructor.
- `Rectangle(const Point &A, const Point &B, const Point &C, const Point &D);` - Constructor that initializes the rectangle with vertices A, B, C, and D.
- `void set_rectangle(const Point &A, const Point &B, const Point &C, const Point &D);` - Sets the vertices A, B, C, and D of the rectangle.
- `Point getA() const;` - Returns the first vertex of the rectangle.
- `Point getB() const;` - Returns the second vertex of the rectangle.
- `Point getC() const;` - Returns the third vertex of the rectangle.
- `Point getD() const;` - Returns the fourth vertex of the rectangle.
- `void translate(const Vector &v);` - Translates the rectangle by the given vector.
- `void symmetry(const Line &k);` - Reflects the rectangle across the given line.

- `double perimeter() const;` - Returns the perimeter of the rectangle.
- `double area() const;` - Returns the area of the rectangle.
- `bool inside(const Point &A);` - Checks if point A is inside the rectangle.
- `bool is_square() const;` - Checks if the rectangle is a square.
- `friend bool operator==(const Rectangle &r1, const Rectangle &r2);` - Checks if two rectangles are equal.
- `friend bool operator!=(const Rectangle &r1, const Rectangle &r2);` - Checks if two rectangles are not equal.

1.8 Class Circle

Attributes:

- `Point O;` - The center of the circle.
- `double r;` - The radius of the circle.

Methods:

- `Circle();` - Default constructor.
- `Circle(const Circle &S);` - Copy constructor.
- `Circle(const Point &S, double r);` - Constructor that initializes the circle with center S and radius r.
- `void set_circle(const Point &O, double r);` - Sets the center and radius of the circle.
- `Point getO() const;` - Returns the center of the circle.
- `double getr();` - Returns the radius of the circle.
- `void translate(const Vector &v);` - Translates the circle by the given vector.
- `void symmetry(const Line &k);` - Reflects the circle across the given line.
- `double perimeter() const;` - Returns the perimeter (circumference) of the circle.
- `double area() const;` - Returns the area of the circle.
- `bool belong(const Point &A);` - Checks if point A lies on the circumference of the circle.

- `bool is_inside(const Point &A);` - Checks if point A lies inside the circle.
- `friend bool operator==(const Circle &r1, const Circle &r2);` - Checks if two circles are equal.
- `friend bool operator!=(const Circle &r1, const Circle &r2);` - Checks if two circles are not equal.