```
//Ishaan Bharal (ixb170930)
Package TieFighter

Main Class
{
    public static void main(String[] args)
    {
        -Initialize Scanners
        -Initialize PrintWriters
        -Declare LinkedList of Payloads
        while(pilot_routes hasNextLine())
        {
            String line = nextLine();
            -Validate with RegEx
            list.addLast(-regex matching for name);
            String[] nums = line.split(" ");
            double[][] coords = new double[16][2];
            for(i in nums.length)
            {
                coords[i] = parseDouble(nums[i]);
            }
            list.getLast().setArea(calculate(coords));
            out.write(list.getLast().getName() + " " + list.getLast().getArea());
        }
        -Close First Scanner
        -Close First PrintWriter

        while(commands hasNextLine())
        {
            String line = nextLine();
            switch(line) -Validate with RegEx
            {
                case "sort":
                    if("area")
                    {
                        if("asc")
                            list.sort("area", 0);
                        else if("desc")
                            list.sort("area", 1);
                    }
                    else if("pilot)
                    {
                        if("asc")
                            list.sort("pilot", 0);
                        else if("desc")
                            list.sort("pilot", 1);
                    }
                    out2.write(line + " " + "Head: " + list.first.item + ", Tail: "
```

```
+ list.last.item);
                    break;
                case "<pilot name>":
                    out2.write(line + " " + list.contains("<pilot name>"));
                    break;
                case "<number>":
                    out2.write(line + " " + list.contains("<number>"));
                    break;
            }
        }
        -Close Second Scanner
        -Close Second PrintWriter
    }

    double calculate(double[][] coords)
    {
        double number = 0;
        for(i = 0; i < coords.length; i++)
            number += (x_i+1 + x_i)(y_i+1 - y_i);
        return 1/2 * |number|;
    }
}

Payload Class - Comparable
{
    String pilot;
    double area;
    boolean flag;

    Payload(String pilot){this.pilot = pilot;}

    int compareTo(E obj)
    {
        return flag? this.area - obj.area : this.pilot.compareTo(obj.pilot);
    }

    public String getPilot(){return pilot;}
    public void setPilot(String pilot){this.pilot = pilot;}
    public double getArea(){return area;}
    public void setArea(double area){this.area = area;}
    public boolean isFlag(){return flag;}
    public void setFlag(boolean flag){this.flag = flag;}
}

LinkedList Class
{
    Node first;
    Node last;
```

```
    LinkedList(){super();}

    LinkedList(Collection c)
    {
        for(i in c)
            addLast(c.get(i));
    }

    String toString()
    {
        -Reference to a toString with Node Parameter
    }

    String toString(Node node)
    {
        if(node == last)
            return null;
        return node.item.toString() + toString(node.next);
    }

    void sort(boolean asc)
    {
        Node node = first;
        while(node != last)
        {
            -Sorting Algorithm
            node = node.next
        }
    }

    boolean contains(E obj)
    {
        Node node = first;
        while(node != last)
        {
            if(node.item = obj)
                return true;
            node = node.next
        }
        return false;
    }

    -Acessors / Mutators
}

Node Class - E
{
```

```
    E item;
    Node next;
    Node prev;

    Node(E obj, Node prev, Node next)
    {
        this.item = obj;
        this.prev = prev;
        this.next = next;
    }

    -Acessors / Mutators
}
```