CS133 Bonus Lab

1. (5 points) Write a logarithmic-time function that finds the top-most point in a convex polygon. The polygon is given as a list of points in CCW order.

```cpp
class Point { double x, y; }
Point topMostPoint(std::vector<Point> P);
```

2. (5 points) Write a function that constructs a kd tree from a set of points \( P \).

```cpp
class KDNode {
    enum {AXIS_X, AXIS_Y} splitAxis;
    double splitCoord;
    struct KDNode *left, *right;
};
KDNode* buildKDTree(std::vector<Point> P);
```

3. (10 points) Given a convex polygon \( P \), write a function that finds the oriented minimum bounding rectangle using a rotating-calipers method.

```cpp
class Rectangle { Point p1, p2, p3, p4; }
Rectangle mbr(std::vector<Point> P);
```

4. (10 points) Given a set of points \( P \) and a distance \( d \), develop a plane-sweep algorithm that finds all pairs of points that are within the distance \( d \). Your implementation should be very similar to the rectangle intersection problem by making imaginary squares of side length \( d \) around each point and find if they intersect. Only the points with overlapping squares will be tested for their distance. For simplicity, you can implement the simple plane-sweep algorithm which does not build any search trees.

```cpp
std::vector<std::pair<Point, Point>> distanceJoin(std::vector<Point> P, double d);
```