public static int my\_getMax(int[] cross\_num, int bridge\_num)

{

int i, max = 0, index = -1;

for (i = 0; i < bridge\_num; i++)

{

if (cross\_num[i] > max)

{

max = cross\_num[i];

index = i;

}

}

return index;

}

public static double my\_min(double a, double b)

{

return a < b ? a : b;

}

public static double my\_max(double a, double b)

{

return a > b ? a : b;

}

public static bool is\_OnSegment(my\_point p, my\_point p1, my\_point p2)

{

if (my\_min(p1.x, p2.x) <= p.x && p.x <= my\_max(p1.x, p2.x))

{

if (my\_min(p1.y, p2.y) <= p.y && p.y <= my\_max(p1.y, p2.y))

{

return true;

}

}

return false;

}

public static int my\_judge\_across(my\_bridge b1, my\_bridge b2)

{

double delt1, delt2, delt3, delt4;

delt1 = (b2.p1.x - b1.p1.x) \* (b2.p2.y - b1.p1.y) - (b2.p1.y - b1.p1.y) \* (b2.p2.x - b1.p1.x);

delt2 = (b2.p1.x - b1.p2.x) \* (b2.p2.y - b1.p2.y) - (b2.p1.y - b1.p2.y) \* (b2.p2.x - b1.p2.x);

if ((delt1 < 0 && delt2 < 0) || (delt1 > 0 && delt2 > 0))

return 0;

if (delt1 == 0)

if (is\_OnSegment(b1.p1, b2.p1, b2.p2))

return 1;

if (delt2 == 0)

if (is\_OnSegment(b1.p2, b2.p1, b2.p2))

return 1;

if ((delt1 == 0) && (delt2 == 0))

{

if (is\_OnSegment(b2.p1, b1.p1, b1.p2) || is\_OnSegment(b2.p2, b1.p1, b1.p2))

return 1;

else return 0;

}

delt3 = (b1.p1.x - b2.p1.x) \* (b1.p2.y - b2.p1.y) - (b1.p1.y - b2.p1.y) \* (b1.p2.x - b2.p1.x);

delt4 = (b1.p1.x - b2.p2.x) \* (b1.p2.y - b2.p2.y) - (b1.p1.y - b2.p2.y) \* (b1.p2.x - b2.p2.x);

if ((delt3 < 0 && delt4 < 0) || (delt3 > 0 && delt4 > 0))

return 0;

if (delt3 == 0)

if (is\_OnSegment(b1.p1, b2.p1, b2.p2) == false)

return 0;

if (delt4 == 0)

if (is\_OnSegment(b1.p2, b2.p1, b2.p2) == false)

return 0;

return 1;

}