```c
#include <conio.h>
#include <graphics.h>
#include <process.h>

void main() {
    int graphicsDriver = DETECT, graphicsMode;
    int i, viewportXMax, viewportYMax, viewportXMin, viewportYMin;
    int lineX1, lineY1, lineX2, lineY2, slope;
    int regionCodeA[4], regionCodeB[4], andingCode[4];
    int clippedX, clippedY;

    clrscr();
    initgraph(&graphicsDriver, &graphicsMode, "c:\\tc\\bgi");

    printf("\nENTER BOTTOM LEFT CO-ORDINATE OF VIEWPORT: ");
    scanf("%d %d", &viewportXMin, &viewportYMax);
    printf("\nENTER TOP RIGHT CO-ORDINATE OF VIEWPORT: ");
    scanf("%d %d", &viewportXMax, &viewportYMin);

    rectangle(viewportXMin, viewportYMin, viewportXMax, viewportYMax);

    printf("\nENTER CO-ORDINATE OF FIRST POINT OF LINE: ");
    scanf("%d %d", &lineX1, &lineY1);
    printf("\nENTER CO-ORDINATE OF SECOND POINT OF LINE: ");
    scanf("%d %d", &lineX2, &lineY2);

    line(lineX1, lineY1, lineX2, lineY2);
    delay(500);

    for (i = 0; i < 4; i++) {
        regionCodeA[i] = 0;
        regionCodeB[i] = 0;
    }

    slope = (lineY2 - lineY1) / (lineX2 - lineX1);

    if (lineX1 < viewportXMin) regionCodeA[3] = 1;
    if (lineX1 > viewportXMax) regionCodeA[2] = 1;
    if (lineY1 < viewportYMin) regionCodeA[1] = 1;
    if (lineY1 > viewportYMax) regionCodeA[0] = 1;

    if (lineX2 < viewportXMin) regionCodeB[3] = 1;
    if (lineX2 > viewportXMax) regionCodeB[2] = 1;
    if (lineY2 < viewportYMin) regionCodeB[1] = 1;
    if (lineY2 > viewportYMax) regionCodeB[0] = 1;
```

```
printf("\nREGION CODE OF FIRST POINT: ");
for (i = 0; i < 4; i++) {
    printf("%d", regionCodeA[i]);
}

printf("\nREGION CODE OF SECOND POINT: ");
for (i = 0; i < 4; i++) {
    printf("%d", regionCodeB[i]);
}

printf("\nANDING: ");
for (i = 0; i < 4; i++) {
    andingCode[i] = regionCodeA[i] && regionCodeB[i];
}

for (i = 0; i < 4; i++) {
    printf("%d", andingCode[i]);
}

getch();

if (andingCode[0] == 0 && andingCode[1] == 0 && andingCode[2] == 0 && andingCode[3] == 0) {
    if (regionCodeA[0] == 0 && regionCodeA[1] == 0 && regionCodeA[2] == 0 && regionCodeA[3] == 0
        && regionCodeB[0] == 0 && regionCodeB[1] == 0 && regionCodeB[2] == 0 && regionCodeB[3] == 0) {

        clrscr();
        clearviewport();
        printf("\nTHE LINE IS TOTALLY VISIBLE AND NO CLIPPING IS REQUIRED ");
        rectangle(viewportXMin, viewportYMin, viewportXMax, viewportYMax);
        line(lineX1, lineY1, lineX2, lineY2);
        getch();
    } else {
        clrscr();
        clearviewport();
        printf("\nLINE IS PARTIALLY VISIBLE ");
        rectangle(viewportXMin, viewportYMin, viewportXMax, viewportYMax);
        line(lineX1, lineY1, lineX2, lineY2);
        getch();

        if (regionCodeA[0] == 0 && regionCodeA[1] == 1) {
            clippedX = lineX1 + (viewportYMin - lineY1) / slope;
            lineX1 = clippedX;
            lineY1 = viewportYMin;
        } else if (regionCodeB[0] == 0 && regionCodeB[1] == 1) {
            clippedX = lineX2 + (viewportYMin - lineY2) / slope;
```

```
        clippedX = lineX1 + (viewportYMin - lineY1) / slope;
        lineX1 = clippedX;
        lineY1 = viewportYMin;
    } else if (regionCodeB[0] == 0 && regionCodeB[1] == 1) {
        clippedX = lineX2 + (viewportYMin - lineY2) / slope;
        lineX2 = clippedX;
        lineY2 = viewportYMin;
    }

    if (regionCodeA[0] == 1 && regionCodeA[1] == 0) {
        clippedX = lineX1 + (viewportYMax - lineY1) / slope;
        lineX1 = clippedX;
        lineY1 = viewportYMax;
    } else if (regionCodeB[0] == 1 && regionCodeB[1] == 0) {
        clippedX = lineX2 + (viewportYMax - lineY2) / slope;
        lineX2 = clippedx;
        lineY2 = viewportYMax;
    }

    if (regionCodeA[2] == 0 && regionCodeA[3] == 1) {
        clippedY = lineY1 + slope * (viewportXMin - lineX1);
        lineY1 = clippedY;
        lineX1 = viewportXMin;
    } else if (regionCodeB[2] == 0 && regionCodeB[3] == 1) {
        clippedY = lineY2 + slope * (viewportXMin - lineX2);
        lineY2 = clippedY;
        lineX2 = viewportXMin;
    }

    if (regionCodeA[2] == 1 && regionCodeA[3] == 0) {
        clippedY = lineY1 + slope * (viewportXMax - lineX1);
        lineY1 = clippedY;
        lineX1 = viewportXMax;
    } else if (regionCodeB[2] == 1 && regionCodeB[3] == 0) {
        clippedY = lineY2 + slope * (viewportXMax - lineX2);
        lineY2 = clippedY;
        lineX2 = viewportXMax;
    }

    clrscr();
    clearviewport();
    printf("\nAFTER CLIPPING: ");
    rectangle(viewportXMin, viewportYMin, viewportXMax, viewportYMax);
    line(lineX1, lineY1, lineX2, lineY2);
    getch();
```

```
            if (regionCodeA[2] == 1 && regionCodeA[3] == 0) {
                clippedY = lineY1 + slope * (viewportXMax - lineX1);
                lineY1 = clippedY;
                lineX1 = viewportXMax;
            } else if (regionCodeB[2] == 1 && regionCodeB[3] == 0) {
                clippedY = lineY2 + slope * (viewportXMax - lineX2);
                lineY2 = clippedY;
                lineX2 = viewportXMax;
            }

            clrscr();
            clearviewport();
            printf("\nAFTER CLIPPING: ");
            rectangle(viewportXMin, viewportYMin, viewportXMax, viewportYMax);
            line(lineX1, lineY1, lineX2, lineY2);
            getch();
        }
    } else {
        clrscr();
        clearviewport();
        printf("\nLINE IS INVISIBLE: ");
        rectangle(viewportXMin, viewportYMin, viewportXMax, viewportYMax);
        getch();
    }

    closegraph();
    getch();
}
```