Report Number 1

**Title :** DDA Line Drawing algorithm with m.

**Objective :** m

**Algorithm :**

Step 1 : Calculate the value of constant

dx=x2-x1; dy=y2-y1; m = |dy/dx|; step = max(dx, dy)

step 2 : if m > 1 goto step 3 otherwise check m>0 then goto step 4 else goto 7.

step 3 : x1 < x2 or x1>x2 assign x\_increment to 1 or -1 correspondingly and

y1 < y2 or y1>y2 assign y\_increment to m or –m.

step 4 : x1 < x2 or x1>x2 assign x\_increment to (1/m) or -(1/m) and

y1 < y2 or y1>y2 assign y\_increment to 1 or -1.

Step 5 : plot the current point (x, y) and update the value of x,y by x\_increment and

Y\_increment correspondingly.

Step 6 : Repeat step 5 number of step times.

Step 7 : Finish

**Program :**

|  |
| --- |
| *// DDA Line Drawing Algorithm*  *void drawLineDDAWithM(int x1, int y1, int x2, int y2) {*  *glColor3f(0,0, 0);*  *int dx = x2 - x1;*  *int dy = y2 - y1;*  *float m = abs((dy+0.0)/dx);*  *float x\_inc = 0, y\_inc = 0;*  *float x = x1, y = y1;*  *int step = max(dx, dy);*  *if(m<=1){*  *x\_inc = (x1<x2)?1:-1;*  *y\_inc = (y1<y2)?m:-m;*  *}else if(m>1){*  *x\_inc = (x1<x2)?(1/m):(-1/m);*  *y\_inc = (y1<y2)?1:-1;*  *}else{*  *cout<<"Invalid value of m : "<<m<<endl;*  *}*  *glBegin(GL\_POINTS);*  *while(step--){*  *glVertex2i(floor(x), floor(y)); // Draw the current point*  *x += x\_inc;*  *y += y\_inc;*  *}*  *glEnd(); // End drawing points*  *glFlush(); // Flush the OpenGL commands*  *}* |

**Input:**

x1=2,y1=2,x2=15,y2=8

**Output :**

|  |  |
| --- | --- |
| dx : 13 dy : 6  m : 0.462 step : 13  (2.00, 2.00) -> (2, 2)  (3.00, 2.46) -> (3, 2)  (4.00, 2.92) -> (4, 2)  (5.00, 3.38) -> (5, 3)  (6.00, 3.85) -> (6, 3)  (7.00, 4.31) -> (7, 4)  (8.00, 4.77) -> (8, 4)  (9.00, 5.23) -> (9, 5)  (10.00, 5.69) -> (10, 5)  (11.00, 6.15) -> (11, 6)  (12.00, 6.62) -> (12, 6)  (13.00, 7.08) -> (13, 7)  (14.00, 7.54) -> (14, 7) | Screenshot 2024-06-02 214707.png |

**Discussion :**In this p

--- ------ ---

Report Number 2

**Title :** DDA Line Drawing algorithm without m.

**Objective :** m

**Algorithm :**

Step 1 : Calculate the value of constant

dx=x2-x1; dy=y2-y1; steps = max(dx, dy)

Step 2 : Calculate the value of x\_increment and y\_increment by

X\_increment = dx/steps; y\_increment = dy/steps;

Step 3 : Plot the current point (x, y) and update the value of x,y by x\_increment and

Y\_increment correspondingly.

Step 4 : Repeat step 3 number of steps times.

Step 5 : Finish

**Program :**

|  |
| --- |
| *// DDA Line Drawing Algorithm*  *// DDA Line Drawing Algorithm*  *void drawLineDDAWithoutM(int x1, int y1, int x2, int y2) {*  *glColor3f(0, 1, 0);*  *int dx = x2 - x1;*  *int dy = y2 - y1;*  *int steps = std::max(abs(dx), abs(dy)); // Calculate the number of steps*  *float xIncrement = dx / (float) steps; // Calculate the x increment*  *float yIncrement = dy / (float) steps; // Calculate the y increment*  *float x = x1, y = y1;*  *glBegin(GL\_POINTS); // Begin drawing points*  *for (int i = 0; i <= steps; i++) {*  *glVertex2i(floor(x), floor(y)); // Draw the current point*  *x += xIncrement; // Increment x*  *y += yIncrement; // Increment y*  *}*  *glEnd(); // End drawing points*  *glFlush(); // Flush the OpenGL commands*  *}* |

**Input:**

x1=2,y1=2,x2=15,y2=8

**Output :**

|  |  |
| --- | --- |
| dx : 13 dy : 6 step : 13  (2.00, 2.00) -> (2, 2)  (3.00, 2.46) -> (3, 2)  (4.00, 2.92) -> (4, 2)  (5.00, 3.38) -> (5, 3)  (6.00, 3.85) -> (6, 3)  (7.00, 4.31) -> (7, 4)  (8.00, 4.77) -> (8, 4)  (9.00, 5.23) -> (9, 5)  (10.00, 5.69) -> (10, 5)  (11.00, 6.15) -> (11, 6)  (12.00, 6.62) -> (12, 6)  (13.00, 7.08) -> (13, 7)  (14.00, 7.54) -> (14, 7)  (15.00, 8.00) -> (15, 7) |  |

**Discussion :**In this p