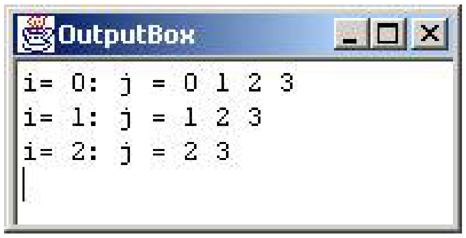
9 Nested Loops and Two-Dimensional Arrays

9.1

The output is



```
i= 0: j = 0 1 2 3
i= 1: j = 1 2 3
i= 2: j = 2 3
```

9.2

```
public void triangles () {
  OutputBox out = new OutputBox();
  out.setSize(200, 200);
  for (int i=-4; i<=4 ; i++) {
    for (int j=0; j<5-Math.abs(i); j++)
      out.print("*");
    out.println();
  }
}
```

9.3

```
public void sawtooth () {
  OutputBox out = new OutputBox();
  out.setSize(200, 300);
  for (int k=1; k<=3; k++)
    for (int i=1; i<= 5; i++) {
      for (int j=1; j<=i; j++)
        out.print("*");
      out.println();
    }
}
```

9.4

```
for (int i=1; i<=n; i++) {
  for (int j=1; j<=m; j++)
     out.print("*");
  out.println();
}
```

9.5

1.

```
int[][] matrix1 (int n) {
  int[][] M = new int[n][n];
  for (int i=0; i<n; i++)
    for (int j=0; j<n; j++)
      M[i][j] = i + j;
  return M;
}
```

2.

```
public int[][] matrix1 (int n) {
  int[][] M = new int[n][n];
  for (int i=0; i<n; i++) {
    for (int j=0; j<n; j++) {
      M[i][j] = Math.min(Math.min(i, n-1-i), Math.min(j, n-1-j));
      System.out.print(M[i][j]);
    }
    System.out.println();
  }
  return M;
}
```

9.6

```
public void addWord (int square, char dir, String word) {
  Point p = findSquare(square);
  if (p == null) {
    new ErrorBox("No such word number: "+square);
    return;
  }
  int row = p.x,
      col = p.y;

  int drow = 0,
      dcol = 1;
  if (dir == 'd') { drow = 1; dcol = 0; }

  for (int i=0; i<word.length(); i++) {

    // Square should be blank or match character
    if (theBoard[row][col] == ' ' ||
        theBoard[row][col] == word.charAt(i)) {
      theBoard[row][col] = word.charAt(i);
      row = row + drow;
      col = col + dcol;
    }
    else // May be trying to write in black square
      if (theBoard[row][col] == '#') {
      new ErrorBox("Wrong length word: "+word);
      return;
    }
    else {  // Must be non-matching character
      new ErrorBox("Non-matching word: "+word);
      return;
    }
  }

  // Check if word filled spaces - if square after
  // word is still on board, but no black, then trouble
  if (theBoard[row][col] != '#') {
    new ErrorBox("Non-filling word: "+word);
    return;
  }
}
```

9.7

```
public void drawLine (Point p1, Point p2) {
  double m = (double)(p2.y - p1.y) / (double)(p2.x - p1.x);
  if (m >= 1)
    drawLineB(p1, p2);
  else if (m >= 0)
    drawLineA(p1, p2);
  else if (m >= -1)
    drawLineC(p1, p2);
  else
    drawLineD(p1, p2);
}

private void drawLineA (Point p1, Point p2) {
  // For any line with a positive slope < 1
  int dx = Math.abs(p1.x-p2.x);
  int dy = Math.abs(p1.y-p2.y);
  int p = 2*dy - dx;
  int x, y, xEnd, xStart;
  if (p1.x > p2.x) {
    xStart = p2.x;
    y = p2.y;
    xEnd = p1.x;
  } else {
    xStart = p1.x;
    y = p1.y;
    xEnd = p2.x;
  }
  setPixel(xStart, y, Color.black);
  for (x=xStart+1; x<=xEnd; x++) {
    if (p < 0)
      p = p + 2*dy;
    else {
      p = p + 2*(dy - dx);
      y++;
    }
    setPixel(x, y, Color.black);
  }
}

private void drawLineC (Point p1, Point p2) {
  // For any line with a negative slope > -1
  int dx = Math.abs(p1.x-p2.x);
  int dy = Math.abs(p1.y-p2.y);
  int p = 2*dy - dx;
  int x, y, xEnd, xStart;
  if (p1.x > p2.x) {
    xStart = p2.x;
    y = p2.y;
    xEnd = p1.x;
  } else {
    xStart = p1.x;
    y = p1.y;
    xEnd = p2.x;
  }
  setPixel(xStart, y, Color.black);
  for (x=xStart+1; x<=xEnd; x++) {
    if (p < 0)
      p = p + 2*dy;
    else {
      p = p + 2*(dy - dx);
      y--;
    }
    setPixel(x, y, Color.black);
  }
}

private void drawLineB (Point p1, Point p2) {
  // For any line with a positive slope > 1
  int dx = Math.abs(p1.x-p2.x);
  int dy = Math.abs(p1.y-p2.y);
  int p = 2*dx - dy;
  int x, y, yEnd, yStart;
  if (p1.y > p2.y) {
    yStart = p2.y;
    x = p2.x;
    yEnd = p1.y;
  } else {
    yStart = p1.y;
    x = p1.x;
    yEnd = p2.y;
  }
  setPixel(x, yStart, Color.black);
  for (y=yStart+1; y<=yEnd; y++) {
    if (p < 0)
      p = p + 2*dx;
    else {
      p = p + 2*(dx - dy);
      x++;
    }
    setPixel(x, y, Color.black);
  }
}

private void drawLineD (Point p1, Point p2) {
  // For any line with a positive slope > 1
  int dx = Math.abs(p1.x-p2.x);
  int dy = Math.abs(p1.y-p2.y);
  int p = 2*dx - dy;
  int x, y, yEnd, yStart;
  if (p1.y > p2.y) {
    yStart = p2.y;
    x = p2.x;
    yEnd = p1.y;
  } else {
    yStart = p1.y;
    x = p1.x;
    yEnd = p2.y;
  }
  setPixel(x, yStart, Color.black);
  for (y=yStart+1; y<=yEnd; y++) {
    if (p < 0)
      p = p + 2*dx;
    else {
      p = p + 2*(dx - dy);
      x--;
    }
    setPixel(x, y, Color.black);
  }
}
```