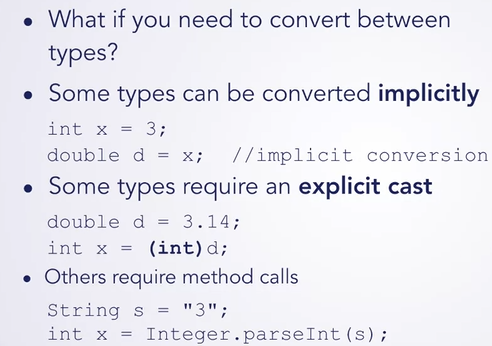
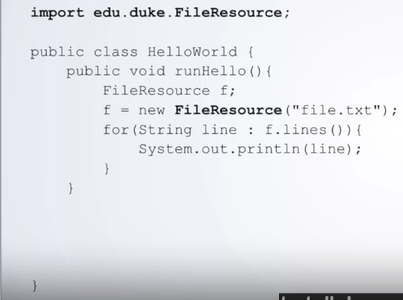


**Clases y Métodos**

Entonces podemos hacer otro punto que tenga sus propios X e Y y represente una instancia diferente del mismo tipo de cosas, otro punto en el plano. Por supuesto, puede crear tantos objetos como necesite para su algoritmo. Una vez que tenga algunos objetos, puede invocar métodos sobre ellos, como p1.distance (p3), que puede considerar como pedirle a p1 que calcule la distancia a p3. Es decir, puede pensar que esta línea de código dice que p1 va a averiguar qué tan lejos está de p3. Puede pensar que el código que se ejecuta para esta llamada al método pertenece lógicamente al objeto

En estos ejercicios, usará las clases de Forma y Punto para responder preguntas sobre una Forma que se compone de una colección de puntos del plano x-y, como se muestra en esta lección. La forma se define dibujando una línea entre dos puntos adyacentes, para cada par de puntos adyacentes, y también una línea entre el primer y el último punto. Asegúrese de consultar la documentación en DukeLearnToProgram para comprender cómo funcionan las clases Shape, Point, DirectoryResource y FileResource: http://www.dukelearntoprogram.com/course2/doc/javadoc/index.html?course=2.

Tarea 1: Calcular información sobre formas

En esta tarea, completará la clase PerimeterAssignmentRunner para calcular muchos datos interesantes sobre las formas. Esta clase se inició para usted en el proyecto BlueJ llamado taskPerimeter (vaya a: http://www.dukelearntoprogram.com/course2/files.php y descargue el proyecto Calculando el perímetro de una forma BlueJ). Este proyecto también contiene varios archivos de datos. Además, deberá consultar la documentación de la clase Shape y la clase Point.

La clase PerimeterAssignmentRunner ya incluye los siguientes métodos completos:

(a) El método getPerimeter tiene un parámetro de tipo Shape. Dada una forma, este método devuelve el perímetro de la forma.

(b) El método testPerimeter no tiene valor de retorno, por lo tanto, su tipo de retorno es nulo. Este método se utiliza para seleccionar un archivo de datos utilizando la clase FileResource, crear una forma basada en los puntos de ese archivo de datos y luego calcular el perímetro de la forma y generar su valor.

(c) El método del triángulo no tiene valor de retorno y crea un triángulo que puede usar para probar los métodos que creará en esta asignación.

(d) El método printFileNames, que discutiremos en la Tarea 2.

(e) El método principal.

Para esta asignación, agregará o modificará varios métodos en la clase PerimeterAssigmentRunner.

1. Complete la escritura del método getNumPoints que tiene un parámetro s que es de tipo Shape. Este método devuelve un número entero que es el número de puntos en Shape s. Sugerencia: deberá iterar sobre todos los puntos en la Forma S y contarlos.

2. Agregue código en el método testPerimeter para llamar a getNumPoints e imprimir el resultado.

3. Complete la escritura del método getAverageLength que tiene un parámetro s que es del tipo Shape. Este método devuelve un número de tipo doble que es el promedio calculado de todas las longitudes de los lados en la Forma S.

4. Agregue código en el método testPerimeter para llamar al método getAverageLength e imprimir el resultado. Tenga en cuenta que si selecciona el archivo example1.txt, la longitud promedio del lado debería ser 4.0.

5. Complete la escritura del método getLargestSide que tiene un parámetro que es de tipo Shape. Este método devuelve un número de tipo doble que es el lado más largo de la Forma S.

6. Agregue código en el método testPerimeter para llamar al método getLargestSide e imprimir el resultado. Tenga en cuenta que si selecciona el archivo example1.txt, el lado más largo debería ser 5.0.

7. Complete la escritura del método getLargestX que tiene un parámetro que es de tipo Shape. Este método devuelve un número de tipo doble que es el valor x más grande sobre todos los puntos en la Forma s.

8. Agregue código en el método testPerimeter para llamar al método getLargestX e imprimir el resultado. Tenga en cuenta que si seleccionara el archivo example1.txt, el lado más largo debería ser 4.0.

Tarea 2

En esta tarea, encontrará el perímetro más grande sobre varias formas examinando varios archivos que representan formas, calculando el tamaño del perímetro más grande y también el nombre del archivo con el perímetro más grande. Agregará nuevos métodos a la clase PerimeterAssignmentRunner.

La clase PerimeterAssignmentRunner ya incluye el siguiente método, debe comprender lo siguiente. El método printFileNames no tiene parámetros ni valor de retorno, por lo tanto, el tipo de retorno es nulo. Este método primero crea un DirectoryResource. Cuando esto sucede, se le solicita que seleccione un archivo o archivos. Puede seleccionar un grupo de archivos juntos haciendo clic en el nombre de un archivo, y luego mantenga presionada la tecla Mayús y seleccione un segundo archivo. Se resaltarán todos los archivos entre el primer y el segundo archivo. El código luego itera sobre todos los archivos que ha seleccionado utilizando un bucle for y el método selectedFiles, imprimiendo el nombre del archivo para cada archivo.

Para esta asignación, agregará o modificará varios métodos en la clase PerimeterAssigmentRunner:

1. Complete la escritura del método getLargestPerimeterMultipleFiles que no tiene parámetros. Este método crea un recurso de directorio (para que pueda seleccionar varios archivos) y luego itera sobre estos archivos. Para cada archivo f, convierte el archivo en un FileResource con la línea

FileResource fr = new FileResource (f);

Luego crea una Forma a partir de FileResource y calcula el perímetro de las formas. ¿Qué más necesita hacer? Debe devolver el perímetro más grande sobre todas las formas en los archivos que ha seleccionado.

2. Termine de escribir el método vacío testPerimeterMultipleFiles para llamar a getLargestPerimeterMultipleFiles e imprimir el perímetro más grande. Este método no tiene parámetros ni valor de retorno. Seleccionará los archivos cuando ejecute este método (sugerencia: consulte nuestra documentación para la clase DirectoryResource).

3. Termine de escribir el método getFileWithLargestPerimeter que no tiene parámetros. Este método debería, como el método getLargestPerimeterMultipleFiles, crear su propio recurso de directorio, excepto que este nuevo método devuelve el archivo que tiene el perímetro más grande, por lo que tiene el tipo de retorno Archivo.

4. Agregue código al método testFileWithLargestPerimeter para llamar a getFileWithLargestPerimeter. Para el archivo que se devuelve, imprima el nombre de ese archivo.

Es importante probar su código con varios archivos.