***Conociendo el proyecto***

1)

2) ***¿Cuántas clases tiene? ¿Cuál es la relación entre ellas?***

Existen 3 clases : SuperCalculadora, Velocidad, Angulo

- “SuperCalculadora” utiliza a “Velocidad”, y “Velocidad” utiliza a “Angulo”.

***¿Cuál es la clase principal? ¿Cómo la reconocen?***

Es SuperCalculadora ya que desde estan se llaman los objetos “Velocidad” y por medio de esta se llama a “Angulo”.

***¿Cuáles son las clases “diferentes”? ¿Cuál es su propósito?***

“Velocidad” y “Angulo” y “SuperCalculadora”

El proposito de “Velocidad” es la encargada de hacer los calculos con la longitud de la velocidad y con el Angulo.

“Angulo” sirve para manejar angulos en radianes o grados.

3) ***Generen y revisen la documentación del proyecto; ¿está completa la documentación de cada clase? (Detalle el estado de documentación de cada clase: encabezado y métodos)***

. Generen y revisen la documentación del proyecto; ¿está completa la documentación de cada clase? (Detalle el estado de documentación de cada clase: encabezado y métodos)

En la clase velocidad no esta completa la documentación, ya que algunos métodos de la clase no están documentados con el objetivo del método o parametros que resive. Estos son restereste,escalar,vectorial. El encabezado de la clase Velocidad esta también sin documentar.Respecto a la clase Angulo tiene documentado su encabezado con el objetivo en general de la clase, junto con documentación para cada clase.

4. ***Revisen el código del proyecto, ¿en qué estado está cada clase? (Detalle el estado de codificación)***

De la clase SuperCalculadora los metodos: adicione, elimine, duplique, calcule(operarion), calcule (operacion, parametro), consulte, ok no estan completos o vacios, ademas el constructor esta vacio.

De la clase Velocidad los metodos componenteH, componenteV, longitud retornan 0.0, el metodo Angulo retorna null, los metodos equals(angulo), equals(object) retornan false, los metodos rote, sume, reste, escalar, vectorial estan vacios, Tambien el constructor esta vacio.

De la clase Angulo todos los metodos estan incompletos: seno, coseno, grados y radianes retornan 0.0, sume, reste, multiplique, divida retornan null, los metodos equals(angulo), equals(object) retornan false y equals(angulo) respectivamente, y el constructor está vacio.

***Ingeniería reversa***

***2. ¿Qué tipos de contenedores tienen sus colecciones? Consulte el API Java 1¿Qué diferencias hay entre ellos?***

Private Stack<Velocidad> operandonos

Consutamos que significa esto:

Deque<Integer> stack = new ArrayDeque<Integer> ();

No hay ninguna diferencia entre ellos porque solo hay uno

***Conociendo Pruebas en BlueJ***

1. ***Revisen el código de la clase AnguloTest. ¿cuáles etiquetas tiene (componentes con símbolo @)? ¿cuántos métodos tiene? ¿cuantos métodos son de prueba? ¿cómo los reconocen?***

Tiene las etiquetas @Before, @Test, @After

Tiene 7 metodos, son 5 de prueba, para reconocerlos basta verificar arriba de cada metodo que aparezca @”algo”, si aparece @Before en un metodo quiere decir que hace este metodo antes de hacer los @Test (es decir los metodos de prueba), @After nos dice que dichos metodos nos funcionan para que funcionen despues de hacer los Tests.

1. ***Ejecuten los tests de la clase AnguloTest. (click derecho sobre la clase, Test All) ¿cuántos tests se ejecutan? ¿cuántos pasan las pruebas? ¿por qué?***

Se ejecutan los metodos con etiqueta @Test que son 5, ninguno para las prubas ya que para esto todas las expresiones (o AssertTrue, AssertFalse…) deben ser verdaderas , si alguna(al menos una) no es correcta no va pasar el Test y por ende mandara error

1. ***Estudie las etiquetas encontradas en 1. Expliqen en sus palabras su significado***

@Before son los metodos que actuan antes de hacer los @Test (es decir los metodos de prueba), @After nos dice que dichos metodos funcionen despues de hacer los Tests.

***4. Estudie los métodos assertTrue, assertFalse, assertEquals, assertNull y fail de la clase Assert del API JUnit 2. Explique en sus palabras que hace cada uno de ellos.***

AssertTrue sirve para evaluar que una expression sea correcta o verdadera

AssertFalse sirve para evualuar que una expression sea falsa

AssertEquals sirve para comparer entre 2 o mas cadenas o expresiones y verficia que estas sean iguales

AssertNull sirve para evaluar que una expression sea nula, es decir que no exista

AssertFails sirve para evaluar que una metodo no funcione bajo ciertas circunstacias

1. ***Investiguen la diferencia que entre un fallo y un error en Junit. Escriba código usando los métodos anteriores para lograr que los siguientes tres casos de prueba se comporten como lo prometen deberiaPasar, deberiaFallar, deberiaErrar.***

El fallo es cuando un programa da respuestas que no son las esperadas, o son incorrectas.

Un error es cuando el programa no compila (por sintaxis, variables no definidas…).

@Test

public void deberiaPasar(){

assertFalse (a0.equals(a0));

assertFalse (a0.equals(new Angulo(0,Angulo.GRADOS)));

assertFalse (a0.equals(a360));

}

@Test

public void deberiaFallar(){

assertTrue (a0.equals(a0));

assertTrue (a0.equals(new Angulo(0,Angulo.GRADOS)));

assertTrue (a720.equals(a0));

assertTrue (aPI.equals(a180));

assertTrue (a180.equals(aPI));

}

@Test

public void deberiaErrar(){

assertFalse (gon400.equals(a0));

assertTrue (gon100.equals(a90));

}