Adiciones al Poryecto:

* Implementacion de la clase output\OutPut.class, se podrá utilizar cuando se finaliza el file de compilar sin ningún error, puede proceder a darle la opción de –o <filename>. Si el file no compilo, puede que no deje utilizarla o lo vuelva archivo pero con el formato <filename>WithErrors.
* En el main aún no se ha implementado la clase lib\ErrorHandler.class. Cuando se termine de aclarar dudas y mejoras en el main, la implementaremos. Debemos preguntar de qué forma y una estrategia para evitar errores.
* No hemos colocado test cases, por el momento solo es CLI interactivo de prueba.
* En cada clase solo instanciamos la función mandada a llamar. Aún no sabemos que hará cada constructor, pero si sabemos que heredara funciones ya que son pasos consecutivos para la compilación.
* Agregamos una opción “exit” para salir del compilador.

Diseño:

Nuestro compilador constara de un main con todas las funciones y subfunciones especificadas en las instrucciones, las cuales mandaran a llamar la instancia de la función o funciones que el usuario ingrese. Solamente implementamos el ArrayIndexOutOfBoundsException. Por si no ingresa nada en los campos requeridos.

Nota: Queremos saber también si se puede hacer un println de todas las funciones en forma de main interactivo y tan solo poner “Ingrese opción”.

Otra forma de hacer nuestro main es cuando se ejecute una operación en específico, ya sea –o, -traget, etc. Se vayan a una subfuncion en donde se mande a llamar el constructor, para reducir el main.

También pensamos en una operación que resetee el compilador y que se re-ejecute solo de desde un inicio.

Para mejorar las versiones de nuestro compilador pensamos también usar una interfaz donde el usuario no necesariamente utilice nuestro compilador desde el cmd.