**Universidad de Guadalajara**

**Centro Universitario de Ciencias Exactas e Ingenierías**

**Taller de programación de sistemas**

**Reporte # 10**

**ALUMNO:**

Oscar Alberto Palomares López Sección D15

Quiñones Nunñez Karla Patricia Sección D07

**Descripción de las variables y funciones utilizadas**

Para esta Décima práctica seguimos empleando las 7 clases pasadas y se agregaron las clases10 y S1

**Variables nuevas de la clase EnsambladorHC12Raw:**

private String codigoFinal; Variable que contrendrá los códigos generados tambien tiene sus set y get.

**Variables nuevas de la clase EnsambladorHC12UI:**

private javax.swing.JTextArea jTextArea7;

private javax.swing.JLabel jLabel8;

Variable que contrendrá los códigos generados para mostrarse en la UI tambien tiene sus set y get.

**Métodos nuevos:**

**Clase Practica9:**

**String buscarEnCOntenidoDeArchivo(String etiqueta, String contenidoArchivoTemporalTxt){ Busca una etiqueta en el contenido del archivo temporal para saber que dirección en código hexadecimal debe tomar para el código s1.**

**Clase Practica10:**

**public static String procesarPasoDos(String contenidoPasoDos, String FileName) { Lee linea por linea el contenido del archivoTemporal y hace validaciones sobre el tipo de operando y código de operación que tiene cada linea, posteriormente dependiendo de que tipo de operando, sea crea el código s1 s0 o s9 mandando a llamar a los métodos de la clase s1 para almacenar los datos de estos registros, cuando termina de analizar las lineas regresa el contenido con los códigos generados en una String.**

**private static String procesarS0(String FileName) { Genera el código s0 cuando encuentra el ORG del archivo a ensamblar y regresa la cadena de este.**

**Clase S1:**

**Varibles: Estas son las variables de un código que se describen en las instrucciones de la práctica**

**private long longitud = 0;**

**private String direccion;**

**private String codigoMaquina;**

**private String tipo;**

**getter y setter de esta clase:**

**public String getDireccion() {**

**public void setDireccion(String direccion) {**

**public String getCodigoMaquina() {**

**public void setCodigoMaquina(String codigoMaquina) {**

**public long getLongitud() {**

**public void setLongitud(long longitud) {**

**public String getTipo() {**

**public void setTipo(String tipo) {**

**Otros Métodos;**

**public String getContenido() { Calcula, arma y regresa el contenido del código a generar usando las instrucciones descritas en la práctica.**

**public String calcularChecksum(String longitud){ Calcula y regresa el checksum del código usando las instrucciones descritas en la práctica.**

* Se puede decir que esta práctica estuvo sencilla ya que sólo había que reutilizar el código de las prácticas anteriores y de la clase Practica9 por lo que el algoritmo de esta práctica solo tiene nuevas funciones secuenciales y locales reutilizando el código ya hecho y utilizando una clase nueva que crea objetos y otra que sólo hace búsquedas y decide que tipo de operando es.