Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas
Organización de Lenguajes y Compiladores 2
Vacaciones de diciembre 2018
Catedrático: Ing. Kevin Lajpop
Tutor académico: Javier Navarro

Hoja de calificación proyecto 2

Coline Teacher

Fecha Calificación: ____/____/_____

Nombre: _____ Registro académico: _____

.

NOTA: Los requerimientos mínimos del proyecto son funcionalidades del sistema que permitirán un ciclo de ejecución básica. En el caso de no cumplir con **uno o más** de los siguientes requerimientos **no se calificará el proyecto y el estudiante tendrá una nota de cero puntos.**

No	Requerimientos mínimos	Cumple completamente	No cumple
1.	Lecciones		
2.	Dummy-Coach		
3.	Pantallas de la Plataforma		
4.	Pantalla de Inicio		
5.	Editor de Texto		
6.	Participar en lección		
7.	Nueva Lección		
8.	Evaluar tareas		
9.	Ingreso de código Coline libremente		
10.	Coline		
11.	Case sensitive		
12.	Sobrecarga de métodos		
13.	Recursividad		
14.	Tipos de dato		
15.	Signos de agrupación		
16.	Comentarios		
17.	Operaciones aritméticas, lógicas y relacionales		
18.	Declaración de variables, objetos y arreglos de una dimension		
19.	Nulo		
20.	Tamaño de un arreglo unidimensional		
21.	Operaciones con cadenas		
22.	Casteos		
23.	Procedimientos, funciones y llamadas.		
24.	Constructor		
25.	Sentencias de transferencia		
26.	Detener		

27.	Retorno		
28.	Sentencias de selección		
29.	Si		
30.	Selecciona		
31.	Sentencias cíclicas		
32.	Lectura de datos de parte del usuario		
33.	Todas las sentencias del formato de código intermedio		
34.	Todas las estructuras en tiempo de ejecución		
35.	Promedio mayor o igual a 61 en evaluaciones presenciales.		
36.	Ejecución del código intermedio		

Ponderación

Los puntos se encuentran repartidos en seis secciones. Cada sección será evaluada sobre cien puntos para facilitar la calificación. El puntaje neto de cada sección se encuentra indicado en la siguiente tabla.

DESCRIPCION	Ponderación	Puntaje Obtenido
1. Componentes de la aplicación	**10**	
2. Debugger	**10**	
3. Documentación	**5**	
4. Funcionalidades de nivel básico	**15**	
5. Funcionalidades de nivel intermedio	**20**	
6. Funcionalidades de nivel avanzado	**30**	
7. Errores	**10**	
TOTAL	**100**	

Nota: Para tener nota completa en cada punto de cada sección deberá tener una salida exacta y esperada caso contrario se asignará nota de cero puntos en la sección evaluada, es decir para tener derecho nota completa la salida esperada debe ser exacta, si existiese una salida aproximada, errónea o no es la esperada será ponderado con nota cero en el punto evaluado.

1 Componentes de la aplicación

En esta sección se calificará que las funcionalidades de la aplicación se hayan implementado correctamente. Para cada punto a calificar se realizarán las acciones indicadas y se deberá comprobar, dentro de la aplicación, que la funcionalidad cumple correctamente con lo esperado.

DESCRIPCION	Ponderación	Puntaje Obtenido
Creación de lecciones	**20**	
Crear lección de tipo Dummy-coach	10	
Crear lección de tipo Pro-coach	10	
Participar en las lecciones	**30**	
Dummy-coach	15	
Pro-coach	15	
Guardar lecciones permanentemente (memoria no volátil)	**10**	
Dummy-coach	5	
Pro-coach	5	
Filtrado por tipo de lecciones en la pantalla principal (ambas)	**10**	
Búsqueda de lecciones	**10**	
Dummy-coach	5	
Pro-coach	5	
Listado de lecciones	**20**	
Dummy-coach	10	
Pro-coach	10	
TOTAL	**100**	

2 Debugger

En esta sección se calificará el componente para permitir la depuración en Coline Teacher.

DESCRIPCION	Ponderación	Puntaje Obtenido
Debugger para la generación de código intermedio	**40**	
Modalidad línea a línea	5	
Siguiente línea	2.5	
Saltar arriba	2.5	
Modalidad con puntos de interrupción	10	
Continuar	5	
Puntos de interrupción	5	

Habilitar en tiempo de ejecución	1	
Deshabilitar en tiempo de ejecución	1	
Resaltado de línea en la que existe punto de interrupción	3	
Modalidad Debugger Automático	5	
Lento	2.5	
Rápido	2.5	
Funcionamiento	20	
Mostrar dinámicamente el código generado	15	
Tabla de símbolos	2	
Consola de Errores	2	
Consola de Salida	1	
Debugger para la ejecución de código intermedio	**60**	
Modalidad línea a línea	5	
Siguiente línea	2.5	
Saltar arriba	2.5	
Modalidad con puntos de interrupción	10	
Continuar	5	
Puntos de interrupción	5	
Habilitar en tiempo de ejecución	1	
Deshabilitar en tiempo de ejecución	1	
Resaltado de línea en la que existe punto de interrupción	3	
Modalidad Debugger Automático	5	
Lento	2.5	
Rápido	2.5	
Funcionamiento	40	
Mostrar estado dinámico del Heap (puntero y contenido)	12	
Mostrar estado dinámico del Stack (puntero y contenido)	12	
Mostrar estado dinámico del SP (puntero y contenido)	12	
Consola de Errores	2	
Consola de Salida	2	
TOTAL	**100**	

3 Documentación

En esta sección se calificará que las funcionalidades de la aplicación se hayan implementado correctamente. Para cada punto a calificar se realizarán las acciones indicadas y se deberá comprobar, dentro de la aplicación, que la funcionalidad cumple correctamente con lo esperado.

DESCRIPCION	Ponderación	Puntaje Obtenido
Manual de usuario	20	
Pasos para crear una lección	4	
Pasos para participar en una lección	4	
Funcionamiento de la página principal	4	
Funcionamiento de traductor a código intermedio	4	
Funcionamiento de debbuger	4	
Manual técnico	30	
Descripción de aplicación y requerimientos mínimos	5	
Diagrama de clases	5	
Diagrama de paquetes o de herencia	5	
Descripción de herramientas utilizadas	5	
Descripción de clases y métodos principales	10	
Gramáticas	50	
Gramática de Coline	25	
Explicación de acciones semánticas	5	
Pregunta sobre la TDS utilizada	10	
Pregunta sobre la gramática	10	
Gramática de código intermedio	25	
Explicación de acciones semánticas	5	
Pregunta sobre la TDS utilizada	10	
Pregunta sobre la gramática	10	
TOTAL	**100**	

4 Funcionalidades de nivel básico

En esta sección se calificará que las funcionalidades básicas del lenguaje Coline hayan sido implementadas correctamente. Los archivos de entrada estarán distribuidos de la siguiente manera:

4.1 Operaciones

En este archivo se evaluarán operaciones aritméticas, relacionales y lógicas. Para las condiciones se utilizará la sentencia sí. El archivo estará separado por funciones que retornan el resultado de las operaciones y este se imprimirá en pantalla.

4.2 Funciones

En este archivo se evaluarán las sentencias mientras, Hacer mientras, para, selecciona y sí. Estas sentencias vendrán anidadas o en secuencia (una después de otra), con condiciones que abarquen los tres tipos de operaciones y con distintos tipos de dato. El archivo estará separado por funciones que retornan el resultado y este se imprimirá en pantalla.

DESCRIPCION	**Ponderación**	**Puntaje Obtenido**
Operaciones	**27**	
Resultado de función operaciones_aritméticas_básicas	2	
Resultado de función operaciones_aritméticas_avanzadas	**3**	
Resultado de función operaciones_relacionales_básicas	**3**	
Resultado de función operaciones_relacionales_avanzadas	4	
Resultado de función operaciones_lógicas_básicas	4	
Resultado de función operaciones_lógicas_avanzadas	5	
Resultado de función operaciones_conjuntas	6	
Funciones	**65**	
Salida de función evaluar_numero	2.5	
Salida de función evaluar_caracter	2.5	
Salida de función saludar	5	
Salida de función ciclo_de_vida	20	
Salida de función factorial_iterativo	10	
Salida de función factorial_recursivo	15	
Salida de función pedir_num	10	
Importar	**2**	
Herencia	**4**	
Modificadores de acceso	**2**	
Total	**100**	

5 Funcionalidades de nivel intermedio

En esta sección se evalúa además del correcto funcionamiento del proceso de compilación, la eficiencia y la resistencia a condiciones no ideales que requieren algoritmos de complejidad alta.

5.1 Tipos de recursión

Esta sección permitirá evaluar la implementación correcta de los ámbitos locales utilizando las estructuras en tiempo de ejecución.

5.1.1 Recursión simple (Utilizando operador ternario)

Una función posee recursividad simple si dentro de su bloque de sentencias solo aparece **una** llamada a sí misma. Esta parte **permitirá la evaluación de la sentencia de selección operador ternario.**

5.1.1.1 Función potencia

La función potencia calculará la siguiente expresión: x^n, de forma recursiva **permitirá evaluar la recursión simple.**

5.1.2 Recursión múltiple

Una función posee recursividad simple si dentro de su bloque de sentencias existe **una o más** llamadas a sí misma.

5.1.2.1 Solución al problema Torres de Hanoi

Las Torres de Hanói es un juego matemático cuya solución es recursiva, esto **permitirá evaluar la recursión múltiple**.

5.1.2.2 Ordenamiento QuickSort

El ordenamiento QuickSort es un algoritmo de ordenación. Este **permitirá evaluar la recursividad múltiple y uso de arreglos**.

5.1.3 Recursión cruzada o mutua

Dos funciones se llaman mutuamente recursivas si la primera función hace una llamada recursiva a la segunda función y la segunda función, a su vez, llama a la primera.

5.1.3.1 Función generatriz de sucesiones Hofstadter

Esta función posee recursividad cruzada ya que lo cual representa una dependencia circular entre dos funciones, esta **permitirá evaluar la recursión cruzada o mutua**

5.1.3.2 Par o impar

Esta parte será un algoritmo recursivo que determine la paridad de un numero n **permitirá la evaluación de la recursividad cruzada**.

5.1.4 Recursión anidada (Ackerman)

La función de Ackerman es una función es altamente recursiva, que posee dentro de sus argumentos llamadas a sí misma. Esta **permitirá evaluar el rendimiento del compilador desarrollado en el grado de recursión más alto**.

5.2 Manejo adecuado de referencias

Se evaluará que el estudiante haya manejado los objetos, las cadenas y los arreglos por referencia.

5.3 Tabla Hash

Este archivo de entrada tendrá las clases necesarias para la implementación de una tabla Hash, la cual permitirá evaluar arreglos de objetos, casteos, ciclos, expresiones de complejidad intermedia.

5.4 Matriz ortogonal

Este archivo de entrada tendrá las clases necesarias para la implementación de una matriz ortogonal, la cual permitirá evaluar objetos, casteos, ciclos, expresiones de complejidad intermedia.

Tiempo estimado para la evaluación de esta sección: **35 minutos.**

DESCRIPCION	**Ponderación**	**Puntaje obtenido**
Tipos de recursión	**30**	
Recursividad simple (Utilizando operador ternario)	2	
Función potencia	1	
Recursividad múltiple	4	
Solución al problema Torres de Hanói	2	
Ordenamiento QuickSort	2	
Recursividad cruzada	9	
Función generatriz de sucesiones Hofstadter	5	
Par o impar	4	
Recursividad anidada	15	
Ackerman	15	
Manejo adecuado de referencias	**10**	
Tabla Hash	**30**	
Salida inserción (código Graphviz)	10	
Salida obtener nombre	10	
Salida eliminación (código Graphviz)	10	
Matriz ortogonal	**30**	
Salida inserción (código Graphviz)	10	
Salida obtener datos de Estudiante	10	
Salida eliminación (código Graphviz)	10	
TOTAL	**100**	

6 Funcionalidades de nivel avanzado

En esta sección se evaluarán estructuras jerárquicas con recorridos recursivos combinando funcionalidades avanzadas de objetos, cadenas y arreglos. Para evaluar la correcta construcción de las estructuras se calificará mediante el código de Graphviz generado, esto permitirá conocer que elementos poseen las estructuras y que los punteros hayan sido implementados correctamente.

6.1 Árbol Binario de Búsqueda

Este archivo de entrada tendrá las clases necesarias para la implementación de un árbol Binario de Búsqueda, el cual permitirá evaluar todos los componentes del lenguaje Coline, incluyendo la colección de estructuras predeterminadas y recorridos recursivos.

6.2 Árbol AVL

Este archivo de entrada tendrá las clases necesarias para la implementación de un árbol AVL, el cual permitirá evaluar todos los componentes del lenguaje Coline, incluyendo la colección de estructuras predeterminadas y recorridos recursivos.

6.3 Árbol B

Este archivo de entrada tendrá las clases necesarias para la implementación de un árbol B, el cual permitirá evaluar todos los componentes del lenguaje Coline, incluyendo la colección de estructuras predeterminadas y recorridos recursivos.

Tiempo estimado para la evaluación de esta sección: **35 minutos.**

DESCRIPCION	Ponderación	Puntaje obtenido
Árbol ABB	**30**	
Salida inserción (código Graphviz)	10	
Salida búsqueda	10	
Salida eliminación (código Graphviz)	10	
Árbol AVL	**30**	
Salida inserción (código Graphviz)	10	
Salida búsqueda	10	
Salida eliminación (código Graphviz)	10	
Árbol B	**40**	
Salida inserción (código Graphviz)	20	
Salida búsqueda	20	
TOTAL	**100**	

7 Errores

7.1 Errores léxicos

Se **evaluará la detección de un error inducido** es importante tener en cuenta que se elegirá aleatoriamente un símbolo que no pertenezca al lenguaje Coline y se debe reportar el mismo.

Para la sección de recuperación se introducirán dos o más errores léxicos separados por el Token centinela para verificar que el estudiante haya implementado la recuperación de errores en modo pánico.

7.2 Errores sintácticos

Se **evaluará la detección de un error inducido** es importante tener en cuenta que se elegirá aleatoriamente un error de estructura que represente un error sintáctico y se debe reportar el mismo.

Para la sección de recuperación se introducirán dos o más errores sintácticos separados por el Token centinela para verificar que el estudiante haya implementado la recuperación de errores en modo pánico.

7.3 Errores en tiempo de ejecución

Se **evaluará la detección de un error inducido**, se debe detener la ejecución y mostrar el reporte de error.

7.3.1 Rango numérico superado

Se definieron rangos para los diferentes tipos de datos y para los arreglos de no cumplirse estos rangos deberá reportarse un error.

7.3.2 División por cero

No es posible la división por cero y por eso deberá reportarse un error

7.3.3 Rango definido superado en arreglos

Los arreglos se declaran con ciertas dimensiones y límites de no cumplirse debe reportarse un error.

7.3.4 Errores de tipo

En una asignación el valor de inicialización posee implícito un tipo diferente al declarado debe reportarse un error

Tiempo estimado para la evaluación de esta sección: **35 minutos.**

DESCRIPCION	Ponderación	Puntaje obtenido
Errores léxicos	**10**	
Detección	2	
Recuperación	8	
Errores sintácticos	**20**	

Detección	5	
Recuperación	15	
Errores en tiempo de ejecución	**70**	
Rango numérico superado (función factorial)	10	
División por cero	10	
Rango definido superado en arreglos	15	
Dimensiones de arreglos	10	
Errores de tipo	10	
Errores en paso de parámetros	15	
TOTAL	**100**	

Restricciones:

- La calificación se realizará sobre archivos ejecutables.
- El proyecto es individual.
- La calificación del proyecto se realizará presencialmente.
- No se puede agregar o quitar algún símbolo en el archivo de entrada. El proyecto deberá funcionar con los archivos que sean proveídos por lo auxiliares para la calificación, sin modificación alguna.
- No será permitido compartir los archivos de entrada durante ni después de la calificación.
- La calificación del proyecto será personal y durará como máximo 120 minutos. Se debe tomar en cuenta que no pueden estar personas ajenas a la calificación, de lo contrario no se calificará el proyecto.
- Anomalías o Copias detectadas de proyectos tendrán de manera automática una nota de 0 puntos y los involucrados serán reportados a la Escuela de Ingeniería en Ciencias y Sistemas, para que se apliquen las sanciones correspondientes.
- Anomalías detectadas en los archivos entregables tendrá de manera automática una nota de 0 puntos, por ejemplo: no se envió código el código correcto, se envió parte del código y no el código completo, archivos ajenos a los entregables del proyecto, no se hizo uso de las herramientas descritas en el enunciado de cada proyecto, entre otras.
- Los archivos de entrada contendrán errores semánticos, sintácticos y léxicos para la verificación de recuperación de errores de la aplicación.
- **Si el incumplimiento de alguna de las anteriores restricciones o bien alguna anomalía detectada durante la calificación los tutores académicos finalizaran la calificación y anularan la nota obtenida hasta el momento, seguido se notificará a los catedráticos y si fuese necesario a la Escuela de Ingeniería en Ciencias y Sistemas**.