

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
ARQUITECTURA DE COMPUTADORES Y ENSAMBLADORES 1
SEGUNDO SEMESTRE 2020
ING. OTTO ESCOBAR
TUTOR ACADÉMICO SECCIÓN A: HERBERTH ARGUETA
TUTOR ACADÉMICO SECCIÓN B: SUSEL RETANA

PRÁCTICA 4

Objetivo General:

- Aplicar los conocimientos adquiridos en el curso sobre el lenguaje ensamblador.

Objetivos Específicos:

- Aplicar el conocimiento de operaciones básicas a nivel ensamblador.
- Conocer el funcionamiento de las interrupciones.
- Comprender el uso de la memoria en los programas informáticos.
- Consolidar los conocimientos de escritura\lectura de archivos.

Descripción:

La práctica consiste en realizar una aplicación en consola utilizando programación a bajo nivel o lenguaje ensamblador, la cual tendrá las funcionalidades de una calculadora simple, en ella se realizarán operaciones aritméticas básicas: suma (+), resta (-), multiplicación (*) y división (/).

Menú Principal:

Este contará con las siguientes opciones:

```
=====
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERIA
CIENCIAS Y SISTEMAS
CURSO: ARQUITECTURA DE COMPUTADORES Y ENSAMBLADORES 1
NOMBRE:
CARNET:

1) CARGAR ARCHIVO
2) CONSOLA
3) SALIR

Escoja Opcion:
=====
```

Cargar Archivo

En esta opción se permitirá ingresar la ruta de un archivo con extensión “json”, el cual contará con la información a procesar.

```
===== CARGAR ARCHIVO =====
INGRESE RUTA: entrada.json
Archivo leído con éxito!
```

Guardará toda la información necesaria en memoria y esperará para poder ser procesada.

Estructura del archivo de entrada JSON

En la siguiente imagen se muestra un ejemplo de un archivo de entrada:

```
{
  "operaciones":
  [
    {
      "operacion1":{
        "mul": {
          "+": {
            "#":4,
            "-": {
              "#":34,
              "div":{
                "#":-30,
                "#":-2
              }
            }
          },
          "#":4
        }
      },
      "operacion2":{
        "add": {
          "/": {
            "sub": {
              "#":-30,
              "#":10
            },
            "#":4
          },
          "id":"operacion1"
        }
      },
      "operacion3":{
        "-":{
          "#":-40,
          "#":-10
        }
      }
    }
  ]
}
```

- Precedencia de operaciones: La precedencia de operaciones ya está definida por la estructura del archivo JSON.
- El archivo de entrada tendrá un único objeto “padre” (en el caso de este ejemplo: “operaciones”), donde se define un array de operaciones.
NOTA: El nombre del objeto padre, puede variar.

- Array de operaciones:
 - Las operaciones contarán con su identificador (en el caso del ejemplo. operacion1, operacion2, operacion3), el cual puede variar. Pueden venir n cantidad de ellas.
 - Las operaciones aritméticas admitidas son:
 - División: Esta se puede declarar con la palabra “div” o el operador “/”.
 - Multiplicación: Esta se puede declarar con la palabra “mul” o el operador “*”.
 - Resta: Esta se puede declarar con la palabra “sub” o el operador “-”.
 - Suma: Esta se puede declarar con la palabra “add” o el operador “+”.

Para las palabras con las que se puede declarar los operadores, es case-insensitive.

- Valores: Los valores a operar pueden ser de dos maneras:
 - Números: Los números serán declarados con el símbolo “#”. (Todos los números pueden tomar un valor de -999 a 999)
 - Resultados de Operaciones: Para poder obtener el resultado de una operación previa, se utilizará la palabra “id” seguido del identificador de la operación que se desea llamar.
ejemplo: “id”:operacion1” //ejecuta operacion1 y devuelve su resultado.
- Solo pueden venir dos operandos por operador. (Puede venir n niveles de operador).
- Se admiten números negativos.

Consola

Se mostrará de la siguiente manera:

```
===== CONSOLA =====
>> |
```

en la cual se podrá ingresar los siguientes comandos

```
show id
show media
show moda
show mediana
show mayor
show menor
exit
```

show estadístico

Donde estadístico podrá tomar los siguientes valores:

media, mediana, moda, mayor, menor.

El resultado de cada uno de estos será calculado tomando en cuenta el resultado de cada una de las operaciones.

Ejemplo:

```
===== CONSOLA =====  
>> show media  
  
Estadístico media: 48  
|
```

show id

Donde id podrá hacer referencia a:

- Operación: Se coloca su identificador y al momento de ejecutar el comando, devuelve el resultado de la operación indicada.

Ejemplo:

```
===== CONSOLA =====  
>> show operacion1  
  
Resultado operacion1: 92  
|
```

- Objeto “padre”: Se coloca su identificador y al momento de ejecutar el comando, creará un archivo en formato JSON con el nombre del identificador del padre (ejemplo: operaciones.json), de la siguiente manera:

```

{
  "reporte":
  {
    "alumno":
    {
      "Nombre":
      "Carnet":
      "Seccion":"8",
      "Curso":"Arquitectura de Computadores y Ensambladores 1"
    },
    "fecha":
    {
      "Dia":1,
      "Mes":3,
      "Año":2019,
    },
    "hora":
    {
      "Hora":1,
      "Minutos":2,
      "Segundos":3
    },
    "resultados":
    {
      "media": 48,
      "mediana": 82,
      "moda": "No hay",
      "menor": -30,
      "mayor": 92
    }
    "operaciones":
    [
      {
        "operacion1":92
      },
      {
        "operacion2":82
      },
      {
        "operacion3":-30
      }
    ]
  }
}

```

La estructura del archivo de salida tiene lo siguiente:

- Datos del alumno: Conformado con el nombre, carnet, sección y nombre del curso.
- Fecha: Mostrará la fecha del día en la que se genero el reporte (día, mes, año).
- Hora: Mostrará la hora en la que se generó el reporte (hora, minutos, segundos).
- Resultados: Mostrará una lista con los siguientes valores, tomando en cuenta el resultado de todas las operaciones, (media, mediana, moda, menor, mayor, varianza, desviación estándar).
- Id Padre: Mostrará un array de las operaciones con su identificador seguido por el resultado de cada una de ellas.

exit

Al ingresar este comando, cerrará la consola y regresará al menú principal.

Salir

Con esta opción, se cerrará el programa y regresará a la consola de DosBox donde quedará listo para compilar otro programa.

Referencias

Estructura JSON:

<https://www.json.org/>

Observaciones y Restricciones:

- **Se realizará de manera individual.**
- **Copias totales o parciales tendrán una nota de 0 y serán reportadas a escuela.**
- El código del programa debe ser estrictamente ensamblador, no se permite el uso de alguna librería.
- El entorno de pruebas a utilizar debe ser DOSBox, el ensamblador a utilizar queda a discreción del estudiante, por ejemplo: MASM, NASM, TASM, FASM, etc.
- El día de la calificación se harán preguntas sobre aspectos utilizados en la elaboración del proyecto, las cuales se considerarán en la nota final.

Requerimientos Mínimos

- Para tener derecho a calificación:
 - Se debe presentar el proyecto en DOSBox.
 - Se debe haber entregado manual de usuario y manual técnico, de lo contrario se asumirá que el estudiante copió.
 - Lectura de archivo.
 - Creación del reporte.
- Enviar archivo con el código utilizado para la práctica y manual técnico antes de las 23:59 horas del Lunes 8 de Octubre de 2020.
 - nombre: [ARQ1]]TP5_#Carnet.rar
 - medio: Classroom