	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I Ernesto Alcántara Concepción

Asignatura: Fundamentos de programación 1122

Grupo: 17

No de Práctica(s): 03

Integrante(s): Asencio Morales Miguel Ángel; Galván Romero Marco Polo; Guadarrama Herrera Ken Bryan; Mendoza Hernández Carlos Emiliano; Sandoval Vásquez Manuel Elihú

No. de Equipo de cómputo empleado:

No. de Lista o Brigada: Equipo 1

Semestre: 1er. Semestre

Fecha de entrega: 6 de octubre del 2021

Observaciones:

CALIFICACIÓN: _____

Práctica 03:
Solución de problemas y algoritmos

Introducción

Un problema informático se puede definir como el conjunto de instancias al cual corresponde un conjunto de soluciones, junto con una relación que asocia para cada instancia del problema, un subconjunto de soluciones (posiblemente vacío).

Para poder solucionar un problema nos apoyamos en la Ingeniería de Software que de acuerdo con la IEEE se define como “la aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software”. Por lo que el uso y establecimiento de principios de ingeniería sólidos, son básicos para obtener un software que sea económicamente fiable y funcione eficientemente.

Por otra parte, un algoritmo se define como un conjunto de reglas, expresadas en un lenguaje específico, para realizar una tarea en general. Es el conjunto de pasos, procedimientos o acciones que permiten alcanzar un resultado o resolver un problema. Estas reglas o pasos pueden ser aplicados un número ilimitado de veces sobre una situación particular.

Una de las etapas de diseño de software, consiste en la creación de un algoritmo, esto con el fin de proponer una o varias alternativas viables para dar solución al problema y con base en esto, tomar la mejor decisión. Para ello, debe analizarse el problema para definirse previamente los conjuntos de entradas y salidas, y entonces resolverlo por medio de algoritmos computacionales.

Un algoritmo es la parte más importante y durable de las ciencias de la computación debido a que este puede ser creado de manera independiente tanto del lenguaje como de las características físicas del equipo que lo va a ejecutar.

Objetivo

- Elaborar algoritmos correctos y eficientes en la solución de problemas siguiendo las etapas de Análisis y Diseño pertenecientes al Ciclo de vida del software.

Actividades

1. A partir del enunciado de un problema, identificar el conjunto de entrada y el conjunto de salida.
2. Elaborar un algoritmo que resuelva un problema determinado (dado por el profesor), identificando los módulos de entrada, de procesamiento y de salida.

Para los problemas, analizar y escribir los problemas, las restricciones, los datos de entrada y los datos de salida, es decir el resultado que se quiere obtener, algoritmo y prueba de escritorio. Escribir si existe alguna restricción en el problema.

1. Diseñar una solución para resolver cada uno de los siguientes problemas y tratar de refinar sus soluciones mediante algoritmos adecuados.

a) Realizar una llamada telefónica desde un teléfono público.

PROBLEMA:

- Realizar una llamada desde un teléfono publico

RESTRICCIONES:

- El teléfono debe funcionar
- Llevar cambio en monedas

ENTRADA:

- Número de teléfono

SALIDA:

- Llamada desde un teléfono publico

SOLUCIÓN:

1. Inicio
2. Buscar un teléfono publico
3. Una vez encontrado, verificar que funcione, si no, volver al paso 2
4. Ingresar el numero de la persona que queremos llamar
5. Ingresar las monedas para que se realice la llamada
6. Si nadie contesta, volver al paso 4 e ingresar otro numero
7. Colgar el teléfono una vez que se agote el tiempo de la llamada

Fin

b) Cocinar una tortilla.

PROBLEMA:

- Cocinar una tortilla

RESTRICCIONES:

- La Tortilla debe ser de maíz o de harina

ENTRADA:

- Masa para hacer tortillas

SALIDA:

- Tortilla cocinada

SOLUCIÓN:

1. Inicio
2. Buscar un área de trabajo adecuada
3. Escoger los ingredientes y utensilios
4. Revolver los ingredientes(amasar)
5. Hacer bolitas con la masa
6. Expandir la masa usando una prensa para tortillas
7. Ponerlas al fuego en un comal
8. Darles vuelta
9. Sacarlas del fuego
10. Ponerlas en un recipiente adecuado

Fin

c) Arreglar un pinchazo de una bicicleta.

PROBLEMA:

- Arreglar el pinchazo de una bicicleta

RESTRICCIONES:

- Llevar herramientas y parches

ENTRADA:

- Rueda de bicicleta pinchada

SALIDA:

- Reparar el pinchazo de la bicicleta

SOLUCIÓN:

1. Colocar la bici en una posición cómoda para quitar la rueda
2. Desmontar la cubierta de la bici
3. Sacar la cámara de la bici
4. Localizar el pinchazo en la cámara
5. Reparar el pinchazo con un parche
6. Montar la cámara y la cubierta
7. Montar la rueda a la bici
8. Colocar la bici en posición y seguir el recorrido

   Fin

d) Hacer palomitas de maíz en una olla puesta al fuego con aceite, sal y maíz.

PROBLEMA:

- Hacer palomitas de maíz

RESTRICCIONES:

- Usar una olla, sal y aceite

ENTRADA:

- Maíz, sal y aceite

SALIDA:

- Palomitas de maíz

SOLUCIÓN:

1. Colocar la olla en la estufa
2. Prender la estufa a fuego medio
3. Agregar aceite a la olla
4. Agregar los granos de maíz palomero a la olla
5. Tapar la olla
6. Cuando se escuchen las explosiones del maíz. Cargar la olla por las agarraderas
7. Agitar suavemente la olla
8. Cuando ya no haya explosiones, apagar el fuego
9. Vaciar las palomitas a un tazón o un recipiente grande
10. Vierte la sal

    fin

e) Cambiar el cristal roto de una ventana.

PROBLEMA:

- Cambiar el cristal roto de una ventana

RESTRICCIONES:

- Que el cristal este roto
- Tener herramientas

ENTRADA:

- Cristal roto de la ventana

SALIDA:

- Reparar el cristal roto de una ventana

SOLUCIÓN:

1. Medir el área completa del cristal
2. Comprar un nuevo cristal con las medidas del anterior
3. Colocar cinta aislante en la parte rota del cristal
4. Quitar con una pala la silicona o mastique de la ventana
5. Retirar los trozos de cristal sin cortarnos
6. Limpiar el área del marco
7. Colocar el cristal nuevo
8. Colocar silicón en todos los bordes
9. Esperar a que seque el silicón
10. Limpiar los posibles residuos del silicón con alcohol

Fin

2. Escribir un algoritmo para:

a) Sumar dos números enteros.

PROBLEMA: Sumar dos números enteros.
RESTRICCIONES: Los números deben ser enteros.

DATOS DE ENTRADA: Dos números enteros.
DATOS DE SALIDA: Un número entero, resultado de la suma de los dos números enteros.
DOMINIO: Todos los números enteros.

SOLUCIÓN:

1. Crear tres variables de tipo entero *sum1, sum2, sum*.
2. Solicitar el primer sumando de tipo entero y guardarlo en la variable *sum1*.
   2.1 Si el tipo de dato ingresado no es entero, regresar al punto 2.
3. Solicitar el segundo sumando de tipo entero y guardarlo en la variable *sum2.*
   3.1 Si el tipo de dato ingresado no es entero, regresar al punto 3.
4. Realizar la operación *sum1* + *sum2* y guardarlo en *sum*.
5. Imprimir *sum*.

PRUEBA DE ESCRITORIO:

Iteración	***sum1***	***sum2***	**Operación**	**Salida**
1	1.2	1.2	-	-
2	5	1.2	-	-
3	5	5	5+5	10

b) Restar dos números enteros.

PROBLEMA: Restar dos números enteros
RESTRICCIONES: Los números deben ser enteros.

DATOS DE ENTRADA: Dos números enteros.
DATOS DE SALIDA: Un número entero, resultado de la resta de los números enteros.
DOMINIO: Todos los números enteros.

SOLUCIÓN:

1. Crear tres variables de tipo entero *min, sus, dif*.
2. Solicitar el minuendo de tipo entero y guardarlo en la variable *min*.

2.1 Si el tipo de dato ingresado no es entero, regresar al punto 2.

3. Solicitar el sustraendo de tipo entero y guardarlo en la variable *sus.*

3.1 Si el tipo de dato ingresado no es entero, regresar al punto 3.

4. Realizar la operación *min* - sus y guardarlo en *dif*.
5. Imprimir *dif*.

PRUEBA DE ESCRITORIO:

Iteración	***min***	***sus***	**Operación**	**Salida**
1	1.2	1.2	-	-
2	7	1.2	-	-
3	7	4	7-4	3

c) Multiplicar dos números enteros.

PROBLEMA: Multiplicar dos números enteros
RESTRICCIONES: Los números deben ser enteros.

DATOS DE ENTRADA: Dos números enteros.
DATOS DE SALIDA: Un número entero, resultado de la multiplicación de los números enteros.
DOMINIO: Todos los números enteros.

SOLUCIÓN:

1. Crear tres variables de tipo entero *fac1, fac2, prod*.
2. Solicitar el primer factor de tipo entero y guardarlo en la variable *fac1*.
   2.1 Si el tipo de dato ingresado no es entero, regresar al punto 2.
3. Solicitar el segundo factor de tipo entero y guardarlo en la variable *fac2.*
   3.1 Si el tipo de dato ingresado no es entero, regresar al punto 3.
4. Realizar la operación *fac1*fac2* y guardarlo en *prod*.
5. Imprimir *prod*.

PRUEBA DE ESCRITORIO:

Iteración	***fac1***	***fac2***	**Operación**	**Salida**
1	1.2	1.2	-	-
2	9	1.2	-	-
3	9	3	9*3	27

d) Dividir un número entero por otro.

PROBLEMA: Dividir un número entero por otro
RESTRICCIONES: Los números deben ser enteros.

DATOS DE ENTRADA: Dos números enteros.
DATOS DE SALIDA: Un número real, resultado de dividir un número entero entre otro.
DOMINIO: Todos los números reales.

SOLUCIÓN:

1. Crear tres variables de tipo entero *dividendo, divisor, cociente*.
2. Solicitar el dividendo de tipo entero y guardarlo en la variable *dividendo*.
   2.1 Si el tipo de dato ingresado no es entero, regresar al punto 2.
3. Solicitar el divisor de tipo entero y guardarlo en la variable *divisor.*
   3.1 Si el tipo de dato ingresado no es entero, regresar al punto 3.
4. Realizar la operación *divisor/dividendo* y guardarlo en *cociente*.
5. Imprimir *cociente*.

PRUEBA DE ESCRITORIO:

Iteración	***dividendo***	***divisor***	**Operación**	**Salida**
1	1.2	1.2	-	-
2	10	1.2	-	-
3	10	4	10/4	2.5

3. Diseñar un algoritmo que visualice y sume la serie de números 3,6,9,12,…,99.

PROBLEMA: Visualizar y mostrar la suma de la serie de números 3,6, 9,12,...99
RESTRICCIONES: Los números deben ser enteros y positivos

DATOS DE ENTRADA: Serie de números 3,6, 9,12,...99
DATOS DE SALIDA: Impresión y resultado de la suma de la serie de números 3,6, 9,12,...99
DOMINIO: Números enteros reales

SOLUCIÓN:

Inicio

1. Empezando por 3, comprobar si este es múltiplo de 3 y está dentro del intervalo 1,100
   1.1 Si esto se cumple, imprimir el numero acompañado del signo (+) y guardarlo en la variable x1,x2,x3...(una variable por cada término).
   1.2 Aumentar en 1 y regresar al paso 1 hasta que se deje de cumplir alguna de ambas condiciones.
   1.3 Una vez que se deje cumplir alguna de ambas condiciones imprimir el signo (=)
2. Calcular e imprimir la suma x1+x2+x3...

Fin

PRUEBA DE ESCRITORIO:

Iteración	Salida
1	3+6+9+12+15+18+21+24+27+30+33+36+39+ 42+45+48+51+54+57+60+63+69+72+75+78+ 81+84+87+90+93+96+99 = 1683

4. Escribir un algoritmo que lea cuatro números y a continuación visualice el mayor de los cuatro.

PROBLEMA:

- A partir de cuatro números enteros, conocer cuáles son mayores a cuatro.

RESTRICCIONES:

- Los números deben ser enteros.
- Solo deben ser visibles los mayores a cuatro.

DATOS DE ENTRADA:

- Cuatro números enteros.

DATOS DE SALIDA:

- De los números ingresados, solo los que sean mayores a cuatro.

DOMINIO:

- Todos los números enteros.

SOLUCIÓN:

1. Crear cuatro variables de tipo entero: num1, num2, num3, num4
2. Solicitar el primer sumando de tipo entero y almacenarlo en num1.
   2.1 Si el número es mayor a 4 entonces imprimir num1.
3. Solicitar el segundo sumando de tipo entero y almacenarlo en num2.
   3.1 Si el número es mayor a 4 entonces imprimir num2.
4. Solicitar el tercer sumando de tipo entero y almacenarlo en num3.
   4.1 Si el número es mayor a 4 entonces imprimir num3.
5. Solicitar el cuarto número sumando de tipo entero y almacenarlo en num4.
   5.1 Si el número es mayor a 4 entonces imprimir num4.

PRUEBA DE ESCRITORIO:

Iteración	num1	num2	num3	num4	Salida
1	1	6	7	9	6 7 9
2	2	4	5	11	5 11
3	4	7	8	10	7 8 10

5. Escribir un algoritmo que cuente el número de ocurrencias de cada letra en una palabra leída como entrada. Por ejemplo, “Moritmer” contiene dos “m”, una “o”, dos “r”, una “t” y una “e”.

PROBLEMA:

- Contar cuantas veces aparece cada letra en una cadena de caracteres.

RESTRICCIONES:

- Los caracteres de la cadena de entrada deben ser letras del abecedario del inglés.

DATOS DE ENTRADA:

- Una cadena de caracteres.

DATOS DE SALIDA:

- Cada letra que aparece en la cadena de entrada, acompañada del numero de veces que aparece en ella.

DOMINIO:

- Todas las letras del inglés, números enteros positivos.

SOLUCIÓN:

1. Solicitar una cadena de caracteres y almacenarla en un arreglo de caracteres llamado *palabra.*
2. Se crea una variable de tipo entero para cada letra del abecedario *a,b,c,…,z,* y se inicializan en 0.
3. Para cada posición de *palabra*, se compara con cada letra del abecedario y se realiza lo siguiente.
    a. Cuando la letra de la cadena es igual a la letra con la que está siendo comparada, se incrementa en uno el valor de la variable de tipo entero con el mismo nombre correspondiente.
4. Para cada variable del abecedario *a,b,c,…,z,* puede ocurrir uno de los siguientes casos.
    a. Si la variable es mayor que cero significa que si aparece en la cadena *palabra*. En este caso, se imprime el número de veces que aparece, es decir, el valor almacenado en esa variable, y la letra a la que corresponde.
    b. Si la variable es igual a cero significa que no apareció en la cadena de entrada, y, por lo tanto, no debe imprimirlo.

PRUEBA DE ESCRITORIO:

Iteración	palabra	palabra[i]	Contador	Salida
1	“Moritmer”	M	m=1	-
2	“Moritmer”	o	o=1	-
3	“Moritmer”	r	r=1	-
4	“Moritmer”	i	i=1	-
5	“Moritmer”	t	t=1	-
6	“Moritmer”	m	m=2	-
7	“Moritmer”	e	e=1	-
8	“Moritmer”	r	r=2	-
9	“Moritmer”	-	-	Tiene 1 e
10	“Moritmer”	-	-	Tiene 1 i
11	“Moritmer”	-	-	Tiene 2 m
12	“Moritmer”	-	-	Tiene 1 o
13	“Moritmer”	-	-	Tiene 2 r
14	“Moritmer”	-	-	Tiene 1 t

Conclusiones

Una vez realizada la práctica, podemos afirmar que es posible la resolución de problemas mediante la aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software siguiendo las etapas de análisis y diseño pertenecientes al ciclo de la vida del software.

Fue interesante plantear la solución de problemas cotidianos desde como calentar una tortilla hasta la multiplicación, división y suma de números mediante la creación de un algoritmo y su verificación en la prueba de escritorio.

Pudimos comprender las etapas del diseño; desde lo principal, como definir los conjuntos de entradas, conjuntos de salidas y el dominio de nuestra solución, así como la descripción de los pasos a seguir, y el uso de diferentes tipos de variables. De igual forma, aprendimos las características que debe cumplir un algoritmo, el cual debe ser preciso, definido, finito, correcto, sencillo, legible, eficiente y eficaz

Consideramos muy eficiente la construcción de algoritmos para obtener alternativas viables para dar solución a cualquier problema con el fin de obtener la mejor opción no tan solo para la construcción de código sino también para tomar la mejor decisión día a día.

Nos fue posible elaborar algoritmos correctos y eficientes para la solución de los problemas planteados, siguiendo las etapas de Análisis y Diseño pertenecientes al Ciclo de vida del software, y su correspondiente prueba de escritorio para verificar nuestras soluciones, por lo tanto, en este punto de la práctica, se concluye que se lograron los objetivos y la práctica se finalizó de manera satisfactoria.