

Guatemala 29 de Octubre del 2022

Universidad Mariano Gálvez Facultad de Ingeniería

Curso: Algoritmos

Ingeniero: Bayron Carranza.

Proyecto Final

Miembros:

Javier Andrés Flores Soto

Lucero Esmeralda Arriaga

Diego Andrés Esquivel Beltetón

Deyler Josue Alejandro Morales Prado

Dylan Alberto Roldan Perez



Índice

Introducción.....	3
Marco teórico.....	4-14
Algoritmo.....	15
Conclusiones.....	16
Recomendaciones.....	17
E-grafía.....	18



Introducción

Detrás de todos los programas informáticos que conocemos y usamos de manera cotidiana para facilitarnos diversas actividades de nuestro día a día, existe todo un proceso para poderlos crear. Este proceso es conocido como programación.

La programación es el proceso de crear un conjunto de instrucciones que le dicen a una computadora cómo realizar algún tipo de tarea, está regularmente se realiza mediante el uso de algoritmos, que se podrían explicar cómo reglas o instrucciones que deben seguirse para resolver el problema y lograr el objetivo.

Gracias a los conocimientos que hemos adquirido a lo largo de este tiempo, hemos creado un sistema de las Elecciones Generales Presidenciales de 2023 mediante un algoritmo, un programa de Pseint y C#, donde hemos utilizado distintas estructuras repetitivas como el for, while y list con el fin de conseguir los resultados de las votaciones.



Marco teórico

1.1 Algoritmo

¿Qué es un Algoritmo en la programación?

Un algoritmo informático es una secuencia de instrucciones finitas que llevan a cabo una serie de procesos para dar respuesta a determinados problemas.

Esto quiere decir que, un algoritmo informático puede resolver problemas a través de un conjunto de instrucciones, están escritas en el lenguaje apropiado según el tipo de Lenguaje de Programación en uso.

Todo Algoritmo consta de tres partes las cuales son:

Entrada: en la entrada o input del algoritmo será donde se introduzcan todos aquellos datos que el algoritmo necesite para operar.

Proceso: con lo recibido en la entrada o input, el algoritmo realizará una serie de cálculos lógicos para resolver el problema.

Salida: los resultados obtenidos en el procesamiento se mostrarán en la salida o output del algoritmo.

¿Para qué sirve?

En las Ciencias de la computación, no obstante, los algoritmos constituyen el esqueleto de los procesos que luego se codificarán y programarán para que sean realizados por el computador.



2.1 Diagrama de flujo

Un diagrama de flujo representa la esquematización gráfica de un algoritmo que muestra de forma gráfica los pasos o procesos a seguir para lograr la resolución de un problema.

La construcción apropiada es extremadamente importante porque a partir del diagrama de flujo se empieza a crear el programa en cualquier lenguaje de programación. Se llaman diagramas de flujo porque los símbolos utilizados están conectados por flechas que indican el flujo o el desarrollo del programa realizado. Para que los diagramas sean comprensibles para todas las personas, los símbolos están normalizados; es decir, se hicieron iconos casi universales, ya que inicialmente cada usuario podía tener los suyos propios.

Los diagramas de flujo son importantes porque nos facilitan expresar visualmente, el flujo de datos a través del sistema de procesamiento de información, en este, se analizan los procesos o trámites que necesitamos realizar plan o un programa.

2.2 Figuras que se utilizan en un diagrama de flujo

Línea de flujo; muestra la dirección del proceso. Cada línea de flujo conecta dos bloques.



Terminación; El símbolo de terminación marca el punto inicial o final del sistema. Por lo general, contiene la palabra "Inicio" o "Fin".

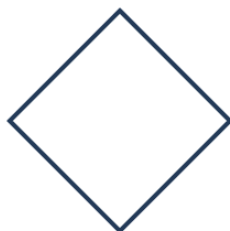




Proceso; Indica un determinado proceso y sus funciones y actividades. En nuestro diagrama de flujo, esta figura se utiliza para leer las variables del programa del proceso de votación como los votos y los porcentajes de los candidatos.



Decisión; Muestra un paso que decide el próximo paso en un proceso. Esta es comúnmente una pregunta de sí/no o verdadero/falso. En nuestro diagrama, lo utilizamos al momento de que el usuario se le realiza la siguiente pregunta; "¿Deseas repetir el proceso? (S/N):" ya que en este paso se determina si el proceso de votación sigue o se detiene, para así sacar el porcentaje total de los votos.



Entrada/Salida; Representa la lectura de datos en la entrada y la impresión de datos en la salida. En nuestro diagrama de flujo utilizamos esta figura al momento en que el usuario selecciona un candidato para votar.





3.1 Pseint

¿Qué es?

PSeInt es la abreviatura de PSeudocódigo Intérprete, una herramienta utilizada principalmente por estudiantes para aprender los fundamentos de la programación y el desarrollo de la lógica.

Propósito de Pseint

PSeInt está pensado para asistir a los estudiantes que se inician en la construcción de programas o algoritmos computacionales. El pseudocódigo se suele utilizar como primer contacto para introducir conceptos básicos como el uso de estructuras de control, expresiones, variables, etc, sin tener que lidiar con las particularidades de la sintaxis de un lenguaje real. Este software pretende facilitarle al principiante la tarea de escribir algoritmos en este pseudolenguaje presentando un conjunto de ayudas y asistencias, y brindarle además algunas herramientas adicionales que le ayuden a encontrar errores y comprender la lógica de los algoritmos.

PSeInt es una herramienta para asistir a un estudiante en sus primeros pasos en programación. Mediante un simple e intuitivo pseudolenguaje en español (complementado con un editor de diagramas de flujo), le permite centrar su atención en los conceptos fundamentales de la algoritmia computacional, minimizando las dificultades propias de un lenguaje y proporcionando un entorno de trabajo con numerosas ayudas y recursos didácticos.

La Estructura Repetitiva Mientras (While)

El while es una estructura que se ejecuta mientras la pregunta de control obtiene una respuesta verdadera, cuando la respuesta a la pregunta de control es falsa esta abandona el ciclo.

Este tipo de estructura es recomendable cuando dentro del programa se desconoce el momento en que se va abandonar el ciclo. Por ejemplo, si necesitamos realizar un programa que solicite números y los sume hasta que el usuario ingrese un número negativo, como no se sabe en que momento el usuario ingresará un valor negativo, la estructura recomendable es el While (Mientras).



La característica principal del While es que este primero pregunta y después hace.

```
1  Proceso elwhile
2      escribir "Ingrese un número"
3      leer numero
4      c<-1
5      Mientras c<=numero Hacer
6          Imprimir "Vamos en el giro ", c
7          c<-c+1
8      Fin Mientras
9  FinProceso
```

Centinela que inicia en 1

Pregunta que controla el bucle

Incrementamos el centinela para permitir el fin del bucle

3.2 Do while

La Estructura Repetitiva Repetir (Do While)

Funciona de igual manera que el While (Mientras), la gran diferencia es que primero hace y después pregunta, y en lugar de abandonar su ejecución al obtener una respuesta falsa en la pregunta de control, lo hace al momento de obtener una verdadera.

```
1  Proceso elrepeat
2      escribir "Ingrese un número"
3      leer numero
4      c<-1
5      Repetir
6          Imprimir "Vamos en el giro ", c
7          c<-c+1
8      Hasta Que c > numero
9  FinProceso
```

La pregunta no se encuentra al principio sino al final

Cambio de la pregunta en el repeat

3.3 For

La Estructura Repetitiva Para (For)

Es una estructura repetitiva que se emplea cuando se conoce cuantos giros debe realizar el ciclo, por ejemplo, si se realiza un algoritmo que le solicite al usuario cuantos números va a sumar, el algoritmo conocería la cantidad de giros a partir de la cantidad de números ingresados por el usuario.

```
1  Proceso elfor
2      escribir "Ingrese un número"
3      leer numero
4      Para c<-1 Hasta numero Con Paso 1 Hacer
5          Imprimir "Vamos en el giro ", c
6      Fin Para
7  FinProceso
```

Incremento que se asigna al centinela

Inicializa el sistema

Límite del ciclo



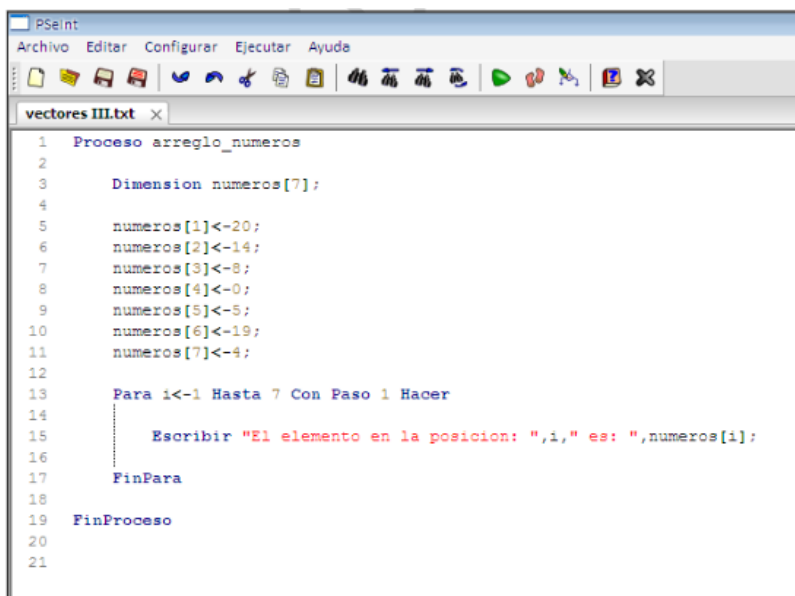
3.4 List (arrays)

Los arreglos son estructuras de datos homogéneas (todos los datos son del mismo tipo) que permiten almacenar un determinado número de datos bajo un mismo identificador, para luego referirse a los mismos utilizando uno o más subíndices. Los arreglos pueden pensarse como vectores, matrices, etc.

Para crear un arreglo en PSeInt se utiliza la palabra clave Dimensión, seguido del nombre del arreglo (identificador) y su tamaño (número de subíndices) entre corchetes [].

Ejemplo:

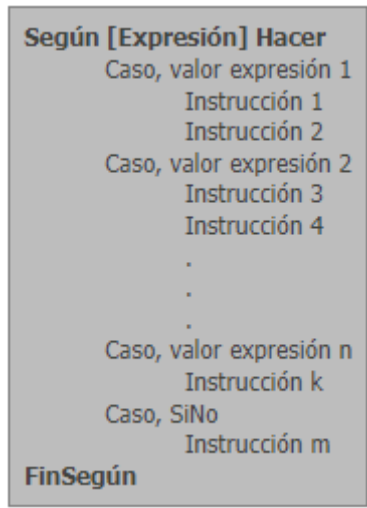
Crear un arreglo llamado números que almacene los siguientes datos: 20, 14, 8, 0, 5, 19 y 4.



```
1  Proceso arreglo_numeros
2
3      Dimension numeros[7];
4
5      numeros[1]<-20;
6      numeros[2]<-14;
7      numeros[3]<-8;
8      numeros[4]<-0;
9      numeros[5]<-5;
10     numeros[6]<-19;
11     numeros[7]<-4;
12
13     Para i<-1 Hasta 7 Con Paso 1 Hacer
14     .....
15         Escribir "El elemento en la posicion: ",i," es: ",numeros[i];
16     .....
17     FinPara
18
19 FinProceso
20
21
```

3.5 Switch

La instrucción Según (Caso) Hacer es una forma de expresión de un anidamiento múltiple de instrucciones Si ... Entonces – SiNo. Su uso no puede considerarse, por tanto, estrictamente necesario, puesto que siempre podrá ser sustituido. La sintaxis será:



Permite trabajar con un sistema de valor numero, donde, según el valor elegido ocurría una secuencia de acciones.

Desarrollo:

Debe existir previamente la variable con la que se trabajará, ya sea por el comando 'Leer', o por 'Asignar'

La variable solo debe ser de valor numérico, no se permite del tipo texto

Para comenzar el comando, este inicia con la palabra 'Según' seguido de la variable numérica previamente existente y seguido de la palabra 'Hacer'.

Ejemplo:

```
Segun variableNumerica Hacer
```

Tras tener el comienzo del comando, prosigue una secuencia de acciones que trabajan con opciones/condiciones numéricas que las llaman:

```
1:
```

```
acciones
```



Las opciones/condiciones numéricas deben ser valores numéricos (1 , 2 , 3 , etc...), estas terminar con un ':' (dos puntos) y seguido de las acciones hacer si esta opción es elegida

1:

acciones

2:

acciones

3:

acciones

Tras finalizar las opciones/condiciones numéricas y sus respectivas acciones, se debe finalizar el comando con un 'Fin Según':

Segun variableNumerica Entonces

opciones y acciones

Fin Segun



4.1 C#

"C#" es un lenguaje de programación multiparadigma desarrollado y estandarizado por la empresa Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA e ISO. C# Es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común. A continuación se explicará el código del programa que se llevó a cabo.

El código de c# está compuesto de varios double para poder marcar los votos de los 5 candidatos esto se hace para así registrar los votos de forma eficiente, también para esto tiene que pasar por un ciclo do y do while para poder hacer que los votos se marquen mediante posiciones. también este ciclo se estará usando en una parte del código para poder repetir el proceso de los votos. Y por último se hace un arreglo para poder seleccionar las posiciones de los candidatos.

4.2 Do while

Definido como un bucle de repetición condicional, se ejecuta el cuerpo del bucle pero no se comprueba hasta que este sea verdad, la condición se comprueba después de cada ejecución

En C# todos los ciclos repiten por verdadero y cortan por falso, es importante que las operaciones se ejecuten como mínimo una vez

Si el cuerpo de ascendencia está compuesto por una única instrucción no es necesario encerrar entre llaves, si existen más de una es necesario y deberán escribirse entre el "do" y el "while" Después del paréntesis que encierra la condición es necesario el punto y coma

<pre>do instrucción 1; while (condición);</pre>	<pre>do { instrucción 1; instrucción 2; ... } while (condición);</pre>
---	--



4.3 For

Principalmente se orienta a los vectores pudiendo modificar, agregar, eliminar o consultar datos que se encuentren según el índice. Por esto último, una condición mínima del vector es que debe ser ordenado, porque si se intenta leer un dato inexistente, esto genera un error de programación.

Debe ser definido hasta el momento de ejecución del bucle y esta se ejecutará hasta que la condición sea verdadera, deberá establecerse cuál es el valor que tomará el índice luego de cada interacción.

```
1 |  
2 | for(int i = 0; i <= 10; i++)  
3 | {  
4 |     Console.WriteLine("Iteración número: " + i.ToString());  
   | }
```

4.4 List

Son una colección de datos del mismo tipo, como los arrays, a diferencia de que en una lista podemos añadir y quitar datos fácilmente y cuando queramos. Palabra reservada List, seguido del tipo de dato que queremos almacenar entre los símbolos mayor y menor que <>.

Permiten almacenar una secuencia de variables, este objeto en C# es tipado, es decir, se le indica que tipo de objeto va a contener (Strings, números o tipo T)

Tipo T es cualquier tipo de objetos definido en el código fuente

Para inicializar una lista en C#, seguido del tipo de dato que va a contener dicha lista entre los símbolos <>. A continuación, debemos utilizar el constructor de List como si de cualquier otro objeto C# se tratase.

```
1 List<string> morePersonNames = new List<string>() {  
2     "Persona 1", "Persona 2", "Persona 3"  
3 };
```



4.5 Switch

Una instrucción switch ejecuta la *lista de instrucciones* en la primera *sección de switch* cuyo *patrón de caso* coincide con una expresión de coincidencia y cuya restricción de caso, de haberla, se evalúe como true. Una instrucción switch evalúa los patrones de casos en el orden de texto de arriba a abajo. El compilador genera un error cuando una instrucción switch contiene un caso inaccesible. Ese es un caso que ya se controla mediante un caso superior o cuyo patrón es imposible de hacer coincidir.

```
int a = 1;

switch (a)
{
    case 1:
    case 2:
    case 3:
        Console.WriteLine("Primero, segundo o tercero");
        break;
    case 4:
        Console.WriteLine("Cuarto");
        break;
    case 5:
        Console.WriteLine("Quinto");
        break;
    default:
        Console.WriteLine("Nada de lo anterior");
        break;
}

Console.ReadKey();
```

AulaFacil.com



Algoritmo

Inicio

- Recolectar los candidatos que participaran en las elecciones.
- Identificar los candidatos por medio de un valor alfanumérico. (**Carga de candidatos**)
- Mostrar a los candidatos por medio de una lista ordenada.
- Registrar los votos por medio del módulo **Carga de votos**.
- Hacer un conteo de votos pertenecientes a cada candidato.
- Mostrar al usuario una lista ordenada con la posición en la que quedaron cada candidato con sus porcentajes.

Final



Conclusiones

- Este trabajo nos ayudó a conocer de una mejor forma el uso de las diferentes herramientas que utilizamos y los diferentes ciclos que hay en cada herramienta.
- A través de la programación se pueden crear todo tipo de programas, desde el más sencillo, hasta el más complejo, es solo de utilizar un poco de nuestro ingenio para poderlo crear, el único límite es nuestra lógica.



Recomendaciones

- Al momento de que el usuario seleccione un candidato, seleccione sólo el número del candidato por el cual desea votar (1-5), esto con el fin de que el programa registre su voto correctamente.
- Usar de manera efectiva la lógica para lograr realizar el código sin ningún tipo de dificultad.
- Investigar los ciclos repetitivos de los programas utilizados para obtener resultados óptimos.



E-grafía

- Anónimo. (7 de Enero de 2017). Smartdraw. Recuperado el 2022 de Octubre de 13, de <https://www.smartdraw.com/flowchart/simbolos-de-diagramas-de-flujo.htm>
- Anónimo. (3 de Agosto de 2016). ZenflowChart. Recuperado el 15 de Octubre de 2022, de <https://www.zenflowchart.com/diagrama-de-flujo-simbologia>
- Anton, C. (23 de Agosto de 2019). WordPress. Recuperado el 14 de Octubre de 2022, de <https://carlosseguraantoncom.home.blog/2019/08/23/que-es-un-algoritmo-informatico/>
- Oliveria, W. (19 de Abril de 2018). Heflo. Recuperado el 13 de Octubre de 2022, de <https://www.heflo.com/es/blog/modelado-de-procesos/significado-simbolos-diagrama-flujo/>
- BillWagner. (22 de 09 de 2022). *Microsoft*. Obtenido de <https://learn.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/>
- BillWagner. (04 de 10 de 2022). *Microsoft*. Obtenido de <https://learn.microsoft.com/es-es/dotnet/csharp/language-reference/statements/selection-statements>
- Anónimo, (2021) Bucle For en C# (Parte 1) <https://clase13.com/2021/10/11/bucle-for-csharp-parte-1/>
- Anónimo, (2022) Listas en C# <https://bugeados.com/programacion/csharp/listas-en-csharp/>
- Anónimo, (24 de enero 2022) Bucle for – Wikipedia, la enciclopedia libre https://es.wikipedia.org/wiki/Bucle_for
- Gil Gómez, JA. (2016). Implementación de bucles en C con do .. while. <http://hdl.handle.net/10251/66676>