NOMBRE DE LA MATERIA: **PROGRAMACIÓN**

NRC: **42555**

HORARIO: **MARTES Y JUEVES 9 AM – 10:55AM**

ESTUDIANTE: **EFRAIN ROBLES PULIDO**

CODIGO: **221350095**

TEMA: **FASES DE LA CREACIÓN DE UN PROGRAMA**

FECHA: **29 de Agosto de 2021**

Fases de la creación de un programa

1. Buscar el objetivo

Es la fase donde se **enuncia el problema u objetivo que se quiere cumplir**. Es decir, qué quieres que haga tu programa. Por lo general no debes ser ambiguo en la definición, debe quedar todo bien explicado.

2. Analizar el problema

Es el análisis de información (datos de entrada) que necesitas para obtener el resultado, la información que se desea producir (datos de salida) y el método para procesar los datos. Básicamente lo que debes definir en esta etapa es, qué necesita tu programa para cumplir el objetivo o resolver el problema.

3. Diseño del algoritmo

Este paso es la parte creativa y vital, de ello dependerá si vas a poder implementar el programa escribiendo código fuente o no. Se puede hacer mediante un diagrama de flujo (gráfico) o mediante pseudocódigo (o ambos).

- **Diagrama de flujo**: Es la forma gráfica de representar un algoritmo**.** Se usan varios símbolos que definen los procesos en la computación. Los símbolos se relacionan con los otros siguiendo un orden mediante unas líneas de flujo.

Se deben de respetar las reglas implementadas por la ANSI:

- **Inicio y final**: se representan con una elipse.
- **Entrada y salida de datos**: se representa por un romboide. La entrada es la lectura de datos y la salida la impresión. En cambio, las fuentes de los datos o los destinos pueden ser diversos. Algunos autores suelen usar un símbolo de impresión…
- **Decisión**: es un rombo que indica comparación de valores.

- **Proceso**: es un recuadro, indicará asignación de un valor en la memoria y/o la ejecución de una operación aritmética/lógica.
- **Conector**: es un círculo con algún símbolo en su interior que indica la forma de conexión que se usa.
- **Líneas de flujo**: son flechas que indican la dirección del flujo que sigue la ejecución del programa. Entrelazan la secuencia de pasos u operaciones que se deben hacer desde el inicio hasta el final.
    - **Pseudocódigo**: Es más bien una combinación entre lenguaje natural (español, sueco,…) y palabras reservadas usadas en lenguajes de programación. Es decir, sería como una especie de borrador del programa. El paso previo a escribir el código fuente. Pero, por supuesto, el pseudocódigo no es entendible por el compilador, solo por nosotros. Como vimos el pseudocódigo, **describe el algoritmo** o diagrama anterior de una forma más sencilla, y más fácil de modificar que el diagrama (en caso de querer rediseñarlo, habría que dibujarlo desde cero prácticamente). El problema es que no tiene normas, y se suelen usar palabras que nos recuerdan al lenguaje de programación, y eso varía de un programador a otro.

4. Codificación

Ahora se transcribe el pseudocódigo o pasar el diagrama a código fuente. Para saber escribirlo, debes conocer a fondo el lenguaje de programación que estés usando. En este caso, sí que es un código comprensible por el compilador o el IDE que estemos usando.

5. Prueba y depuración

Una vez el programa ya está creado, deberíamos compilarlo para pasarlo a un formato comprensible por la máquina, es decir, a binario. Una vez tenemos el ejecutable, puedes probarlo para ver si funciona adecuadamente. Puedes usar un proceso de **debugging o depuración** para localizar posibles errores o vulnerabilidades de tu programa.

Cuando compilas, es el propio compilador el que suele mostrar **mensajes de advertencia o errores** que detienen el proceso de compilación. Normalmente son cosas no declaradas, que has olvidado algún símbolo como los ; en alguna función, errores sintácticos de cualquier tipo, etc. Cuando el código está correcto y compila, eso no quiere decir que el programa esté libre de errores.

Puede haber **bugs** que sucedan solo en determinadas circunstancias, vulnerabilidades que afecten a la seguridad, desbordamientos, etc. Para eso se emplea la depuración, para detectar este tipo de problemas lógicos que son complicados de detectar y que el compilador no es capaz de detectar.

6. Documentación

Es la guía o comunicación escrita es sus variadas formas, ya sea en enunciados, procedimientos, dibujos o diagramas.

A menudo un programa escrito por una persona es usado por otra. Por ello la documentación sirve para ayudar a comprender o usar un programa o para facilitar futuras modificaciones (mantenimiento).

La documentación se divide en tres partes:

1. Documentación Interna
2. Documentación Externa
3. Manual del Usuario.

7. Mantenimiento

Se lleva a cabo después de terminado el programa, cuando se detecta que es necesario hacer algún cambio, ajuste o complementación al programa para que siga trabajando de manera correcta. Para poder realizar este trabajo se requiere que el programa este correctamente documentado.

Bibliografía

Programación: etapas de la programación | ArchiTecnologia. (2021). Consultado el 28 de agosto de 2021 en https://architecnologia.es/programacion-etapas-programacion

tomo, a., & Perfil, V. (2021). Fases para la creación de un programa. Consultado el 28 de agosto de 2021 en http://prograinformatica.blogspot.com/p/fases-para-la-creacion-de-un-programa.html