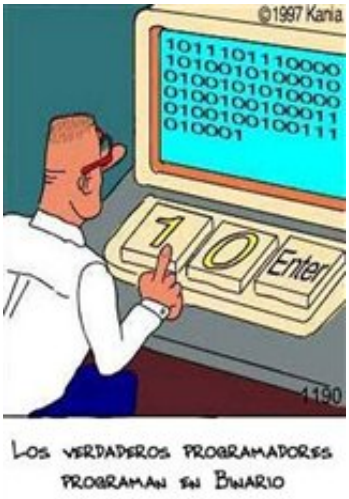
Cuestionario básico de programación

1. **Concepto de Programación de computadoras**

La programación es un proceso por el cual se escribe (en un lenguaje de programación), se prueba, se depura y se mantiene el código fuente de un programa informático. En conceptos más simples la programación de computadoras es el arte de hacer que una computadora haga lo que nosotros queramos.

2. **¿Qué es el diseño de un programa?**

El diseño de un programa es el proceso metodológico el cual es llevado por un programador para realizar el análisis lógico e instrucciones que se llevará a cabo durante la realización de un programa.



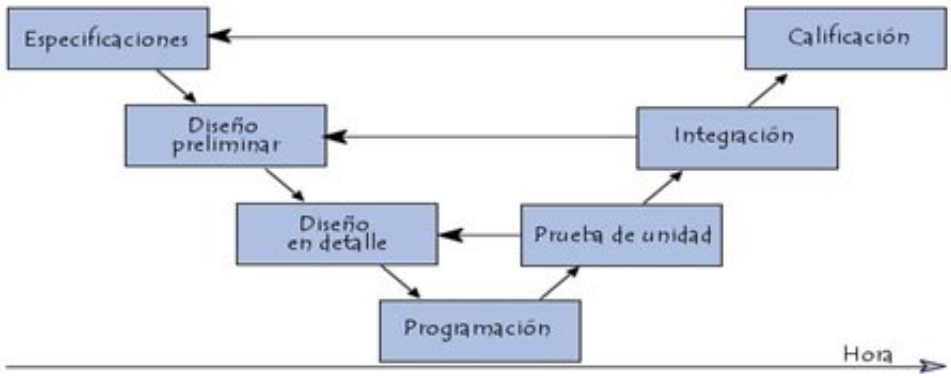
3. **¿Explique el ciclo de vida de una aplicación informática?**

El término ciclo de vida de una aplicación informática describe su desarrollo, desde la fase inicial hasta la fase final. El ciclo de vida básico de una aplicación consta de los siguientes procedimientos:

- **Definición de objetivos**: definir el resultado del proyecto y su papel en la estrategia global.
- **Análisis de los requisitos y su viabilidad**: recopilar, examinar y formular los requisitos del cliente y examinar cualquier restricción que se pueda aplicar.
- **Diseño general**: requisitos generales de la arquitectura de la aplicación.
- **Diseño en detalle**: definición precisa de cada subconjunto de la aplicación.
- **Programación** (programación e implementación): es la implementación de un lenguaje de programación para crear las funciones definidas durante la etapa de diseño.

- **Prueba de unidad**: prueba individual de cada subconjunto de la aplicación para garantizar que se implementaron de acuerdo con las especificaciones.
- **Integración**: para garantizar que los diferentes módulos se integren con la aplicación. Éste es el propósito de la *prueba de integración* que está cuidadosamente documentada.
- **Prueba beta** (o *validación*), para garantizar que el software cumple con las especificaciones originales.
- **Documentación**: sirve para documentar información necesaria para los usuarios del software y para desarrollos futuros.
- **Implementación**
- **Mantenimiento**: para todos los procedimientos correctivos (mantenimiento correctivo) y las actualizaciones secundarias del software (mantenimiento continuo).

El orden y la presencia de cada uno de estos procedimientos en el ciclo de vida de una aplicación dependen del tipo de modelo de ciclo de vida acordado entre el cliente y el equipo de desarrolladores.



Un modelo del ciclo de vida de una aplicación

4. **¿Qué es una metodología de programación?**

Una metodología de programación es un conjunto o sistema de métodos, principios y reglas que permiten enfrentar de manera sistemática el desarrollo de un programa que resuelve un problema algorítmico. Estas metodologías generalmente se estructuran como una secuencia de pasos que parten de la definición del problema y culminan con un programa que lo resuelve.

Los pasos de la metodología consisten en:

- **El Diálogo:** Con la cual se busca comprender totalmente el problema a resolver.
- **La Especificación:** Con la cual se establece de manera precisa las entradas, salidas y las condiciones que deben cumplir.
- **Diseño:** En esta etapa se construye un algoritmo que cumpla con la especificación.
- **Codificación:** Se traduce el algoritmo a un lenguaje de programación.
- **Prueba y Verificación:** Se realizan pruebas del programa implementado para determinar su validez en la resolución del problema.

5. **¿Explique las partes principales de un programa?**

Las partes principales de un programa son:

- **Cabecera (título)**: es una sección obligatoria, debe figurar en todos los programas. Debe comenzar con la palabra reservada program seguida del nombre del programa y un ";". Con esto ya se cumplirían los requisitos mínimos que debe tener una cabecera, pero se puede y es muy recomendable incluir también un comentario.

Las declaraciones: que se clasifican en:

- **Declaración de unidades**: especifica el nombre o identificador de las unidades que se van a utilizar en el programa. Ejemplo: Uses Crt, Dos;
- **Declaración de constantes**: Las constantes son datos que no cambian durante la ejecución del programa y que se definen durante el tiempo de compilación. Ejemplo: Sintaxis: CONST Nombre_Constante = Expresion_1;
- **Declaración de tipos de datos**: La declaración de un tipo de dato consta del nombre o identificador del tipo de dato seguido de los valores que pueden tomar los datos de ese tipo. Por otro lado, existe la posibilidad de que algunos tipos puedan ser subconjuntos o subrangos de otros tipos. También es necesario declarar estos tipos de datos.
- **Declaración de variables**: se reserva espacio en memoria para almacenar los valores que va tomando dicha variable durante la ejecución del programa. La cantidad de memoria reservada dependerá del tipo de variable.
- **Declaración de subprogramas**: es como su cabecera, pero terminada en ";" en vez de con la palabra "is", para expresar que lo que se está dando es una vista de un subprograma cuya definición se haya en otro lugar.

```
static void Main(string[] args)
{
    //Declaracion de variables: Sintaxis -> TipoDato nombreVariable;
    char myChar = 'P';
    byte myByte = 123;
    short myShort = 32767;
    int myInt = 123456789;
    long myLong = 1234567890123456789L;
    float myFloat = 3.1415959F;
    double myDouble = 3.1415959;
     /*Declarar Varias Variables en una misma linea
     * Siempre y cuando sea en la misma linea
     */
    int int1, int2, int3;
    //Asignacion individual
    int1=1;
    int2=int1;
    int3=int1 + int2; //Suma de dos variables
    Console.WriteLine("***Declaracion de Variables***");
    Console.WriteLine("Char: " + myChar);
    Console.WriteLine("Byte: " + myByte);
    Console.WriteLine("Short: " + myShort);
    Console.WriteLine("Int: " + myInt);
    Console.WriteLine("Long: " + myLong);
    Console.WriteLine("Float: " + myFloat);
    Console.WriteLine("Double: " + myDouble);
    Console.WriteLine("int1: " + int1 + "\t int2: "+int2+"\t int3: "+int3);
    //Asignacion multiple
    int1 = int2 = int3;
    Console.WriteLine("int1: " + int1 + "\t int2: "+int2+"\t int3: "+int3);
}
```

- **Cuerpo del programa**: También se le llama bloque del programa, y es junto con la cabecera, la única sección obligatoria en un programa Pascal. Debe comenzar y finalizar con las palabras reservadas begin y end respectivamente.

6. **¿Qué es una estructura general de un programa?**

La estructura exacta de un programa depende del lenguaje que utilicemos y el entorno en el cual lo creemos. Sin embargo, hay algunos principios generales:

- Un cargador - todo programa necesita ser cargado en la memoria por el sistema operativo. De esto se encarga el intérprete.
- Definición de los datos - la mayoría de los programas operan con datos y por lo tanto en el código fuente debemos definir que tipo de datos vamos a utilizar en el programa. Esto se realiza de manera diferente en los distintos lenguajes. Todos los lenguajes que usaremos tienen la posibilidad de crear una nueva definición de datos simplemente al utilizar los datos.
- Instrucciones - son la parte central del programa. Las instrucciones manipulan los datos que hemos definido, realizan cálculos, muestran los resultados, etc.

7. **Defina el concepto pseudocódigos**

```
En-caso-de <expresión> haga
  Caso <opción 1>:
        <instrucciones>
  caso <opción 2>:
        <instrucciones>
  caso <opción 3>:
        <instrucciones>
...
  caso <opción N>:
       <instrucciones>
SINO <instrucciones a realizar si no
       se ha cumplido Ninguna de
       las condiciones anteriores>
Fin-Caso
```

El pseudocódigo (falso lenguaje) es una descripción de alto nivel de un algoritmo que emplea una mezcla de lenguaje natural con algunas convenciones sintácticas propias de lenguajes de programación, como asignaciones, ciclos y condicionales. Es utilizado para describir algoritmos en libros y publicaciones científicas, y como producto intermedio durante el desarrollo de un algoritmo. El pseudocódigo está pensado para facilitar a las personas el entendimiento de un algoritmo, y por lo tanto puede omitir detalles irrelevantes que son necesarios en una implementación.