Bachillerato Internacional

Portada de la Monografía

Los alumnos deben llenar esta hoja y entregarla al supervisor junto con la versión final de su monografía.			
Número de convocatoria del alumno			
Nombre y apellido(s) del alumno			
Número del colegio			
Nombre del colegio			
Convocatoria de exámenes (mayo o noviembre)	MAYO	Año	2013

Asignatura del Programa del Diploma en la que se ha inscrito la monografía: Informática

(En el caso de una monografía en lenguas, señale si se trata del Grupo 1 o el Grupo 2.)

Título de la monografía: Es recomendable mejorar la experiencia de una interfaz de usuario de un programa existente que maneja el almacenamiento de productos de una empresa

Declaración del alumno

El alumno debe firmar esta declaración; de lo contrario, es posible que no reciba una calificación final.

Confirmo que soy el autor de este trabajo y que no he recibido más ayuda que la permitida por el Bachillerato Internacional.

He citado debidamente las palabras, ideas o gráficos de otra persona, se hayan expresado estos de forma escrita, oral o visual.

Sé que el máximo de palabras permitido para las monografías es 4.000, y que a los examinadores no se les pide que lean monografías que superen ese límite.

Esta es la versión final de mi monografía.

Firma del alumno: Fecha:

Informe y declaración del supervisor

El supervisor debe completar este informe, firmar la declaración y luego entregar esta portada junto con la versión final de la monografía al coordinador del Programa del Diploma.

Nombre y apellido(s) del supervisor [MAYÚSCULAS]:

Si lo considera adecuado, escriba algunos comentarios sobre el contexto en que el alumno desarrolló la investigación, las dificultades que encontró y cómo las ha superado (ver página 13 de la guía para la monografía). La entrevista final con el alumno puede ofrecer información útil. Estos comentarios pueden ayudar al examinador a conceder un nivel de logro para el criterio K (valoración global). No escriba comentarios sobre circunstancias adversas personales que puedan haber afectado al alumno. En el caso en que el número de horas dedicadas a la discusión de la monografía con el alumno sea cero, debe explicarse este hecho indicando cómo se ha podido garantizar la autoría original del alumno. Puede adjuntar una hoja adicional si necesita más espacio para escribir sus comentarios.

- Existió un poco de confusión en el planteamiento o formulación del problema, pero con el transcurrir de la investigación, se fueron aclarando las inquietudes.
- Predisposición adecuada para el trabajo.
- Mostró interés en el desarrollo del software.
- El programa fue considerado por el como un tema de actualidad y de mucha dedicación.

El supervisor debe firmar esta declaración; de lo contrario, es posible que no se otorgue una calificación final.

He leído la versión final de la monografía, la cual será entregada al examinador.

A mi leal saber y entender, la monografía es el trabajo auténtico del alumno.

He dedicado [3] horas a discutir con el alumno su progreso en la realización de la monografía.

Firma del supervisor

Fecha:

Formulario de evaluación (para uso exclusivo del examinador)

Criterios de evaluación	Nivel de logro				
	Examinador 1	Máximo	Examinador 2	Máximo	Examinador 3
A Formulación del problema de investigación		2	0	2	
B Introducción		2	0	2	
C Investigación		4	0	4	
D Conocimiento y comprensión del tema		4	1	4	
E Argumento razonado		4	1	4	
F Aplicación de habilidades de análisis y evaluación apropiadas para la asignatura		4	0	4	
G Uso de un lenguaje apropiado para la asignatura		4	1	4	
H Conclusión		2	0	2	
I Presentación formal		4	0	4	
J Resumen		2	1	2	
K Valoración global		4	0	4	
Total (máximo 36)			4		

Informática

Es recomendable mejorar la apariencia de una interfaz de usuario de un programa existente que maneja el almacenamiento de productos de una empresa?

Mayo 2013

Resumen

Introducción:

La idea surgió tras ver que algunos programas eran de una difícil o complicada manipulación por lo cual no todas las personas podían usarlos, para poder usar estos programas tenían que recibir indicaciones de alguien que conozca el programa, por esta razón e decidido ver si al mejorar la apariencia del programa este va a ser de un uso mas simple, ya que las funciones van a estar mas claras y el que lo use va unicamente a tener que fijarse en las cosas que el programa presenta para poder utilizarlo.

Por estas razones es por lo cual e decidido realizar esta monografia con base a ese tema, mi objetivo es poder ver si al hacer esto la gente sin una gran preparación en el manejo de programas pueden usarlos con facilidad.

Resumen de todo lo realizado

Para poder sacar una conclucion de esta investigación se escogió un programa que usa una empresa, este programa se lo va a re-diseñar en lo que concierne a su forma visual según las especificaciones que el usuario final ha indicado en la reunión que se a tenido con el mismo. Dado a que el programa que utiliza esta empresa es privado lo que se ara es que se tomara como ejemplo a este programa pero, se diseñara o creara un nuevo programa con las mismas funciones, es decir que tendrán el mismo proposito pero su diseño en lo que es la apariencia serán distintos.

Para poder hacer esto se a tenido reuniones con el usuario que va a manejar el programa este dio indicaciones de que funciones son necesarias que el programa posea, después de saber que es lo necesario para hacer el programa se a buscado e investigado el como poner una solución a cada objetivo que indico el usuario, ya sabiendo el como se va ha realizar el programa se empezó con la creación de los prototipos, estos prototipos fueron echos con las indicaciones que el usuario indico, ya con los prototipos se empezaron a hacer las pruebas con varias personas (las pruebas se hicieron con 10 personas que no conocían ninguno de los dos programas), la prueba consistía en hacer que hagan tareas que el usuario les indicaba, las tareas debían hacerlas en los dos programas y ellos después de hacerlas indicaban en cual de los dos les pareció mas fácil realizar las tareas, la conclucion que se saco de esta investigación es que a las 10 personas les pareció mas sencilla la manipulación de programa modificado, esto nos comprobó que el hecho de que un programa indique de forma sencilla todas las funciones hace mas fácil su manipulación.

cuerpo de la monografia

Datos de la empresa en la cual se realizara la prueba:
el nombre de la empresa no se ha permitido dar al igual que el nombre del usuario del programa, el nombre del programa al cual se le modificara su forma visual o apariencia para la prueba igualmente no me esta permitido dar.

En esta empresa se usa un programa echo para tener un registro de ventas junto con un catalogo de ropa que vende la empresa en sus distintas sucursales, lo que se esta buscando con esta investigación es ver si al mejorar la forma visual del programa, es decir que en vez de tener que memorizar los códigos que cada prenda posee para poder buscarla en el catalogo, seria mejor tras la modificación del programa organizar las prendas por categorías y géneros por ejemplo: pijamas hombres, pijamas mujeres, entre otras.

Después de hablar con el usuario final de este programa se vio los cambios que este desea hacer para un mejor manejo del sistema con el que trabaja las peticiones no se basaron solo para la tienda principal que tienen sino que lo que desean es que todas sus tiendas puedan estar enlazadas por este programa, esta petición fue echa por el administrador de la empresa.

- Peticiones:
    1. Que el programa posea un botón o un enlace directo para cada opción
    2. Que el catalogo de las prendas este dividido por categorías y que cada una de estas categorías sea mas especifica según su tema correspondiente
    3. Que tenga una función de búsqueda, esta búsqueda si se realizara a través de los códigos de cada prenda
    4. Que cada prenda tenga una información detallada de si misma. Por ejemplo (las tallas, modelos y los colores en los cuales viene y el genero al que pertenece)
    5. Que se pueda editar la información de las prendas (esto es dado a que la empresa modifica los modelos de algunas prendas al igual que las tallas y su gama de colores)
    6. Que se pueda eliminar los registros de las prendas que el usuario desee
    7. Que tenga una función de registro de datos para poder ingresar nuevas prendas al catalogo
    8. Que el usuario principal del programa sea el único con capacidad de dar cambios de los productos
    9. Que el programa tenga su propio sistema de almacenamiento de datos
    10. Que los cambios que realice el usuario principal se hagan también en todos los registros de todas las computadoras que posee la empresa.

- Objetivos de logro
    1. Realizar el programa con los detalles que el usuario solicito
    2. Hacer que los cambios que haga el usuario principal se hagan en todas las computadoras a través del enlace que ya posee la empresa
    3. El sistema de guardado sera especifico y especial para el programa
    4. Tener en cuenta las funciones de registro, búsqueda, editacion y eliminación de los datos del programa

(Los objetivos de logro se los ha puesto en menor cantidad que las peticiones ya que en un solo objetivo esta el punto de cumplir con varias de las peticiones).

Desarrollo del programa:

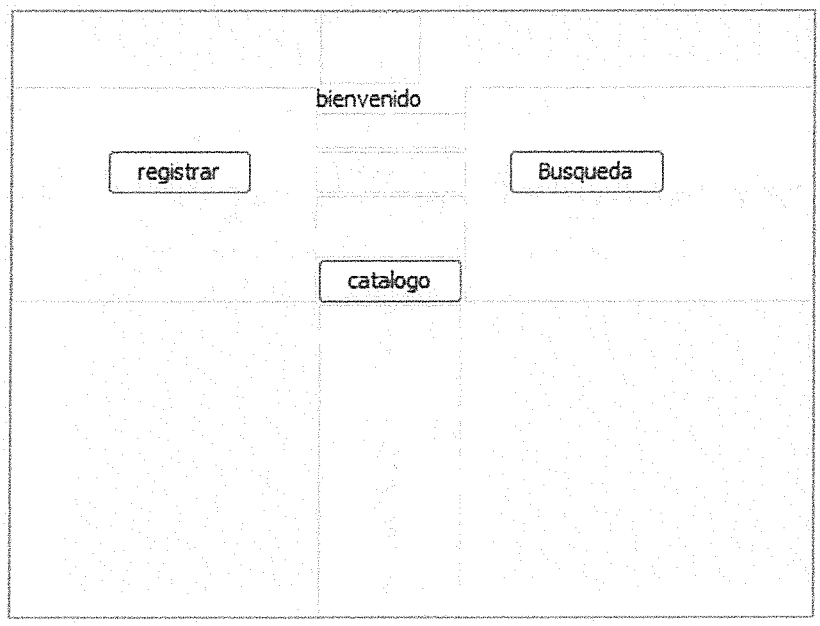
Para el desarrollo de este programa se debe tener en cuenta que existen dos variables las cuales son: el usuario y el programa principal o programa base.

Para realizar los prototipos del programa que se va a editar se utilizara la herramienta de programación llamada "NETBEANS" esta herramienta servirá para realizar el código y sera utilizada para realizar el la editacion del programa anterior tanto en su apariencia es decir su forma visual y ayudara para realizar el código a este nuevo programa.

Para el desarrollo del nuevo programa no se editara en forma directa al programa anterior, sino que este sera tomado como una muestra o ejemplo y lo que se hará es crear un programa nuevo pero que posea características similares al anterior, lo que se cambiara sera limitado unicamente a su forma visual el resto de funciones llevaran exactamente lo mismo que el programa inicial, el objetivo es crear una entrada mas sencilla a cada opción que ofrezca el programa.

La opción de edición de datos unicamente estará disponible para el usuario principal, el cual tendrá un código de acceso para todo el programa y todos sus registros.

Los códigos que se ha utilizado para realizar el programa se los mostrara a continuación (serán fragmentos o partes del código, no se indicara el código completo) con una explicación de el echo en que consiste cada uno, también se mostrara unas capturas de pantalla, el código se pondrá con su pantalla del programa en forma respectiva:



Código:

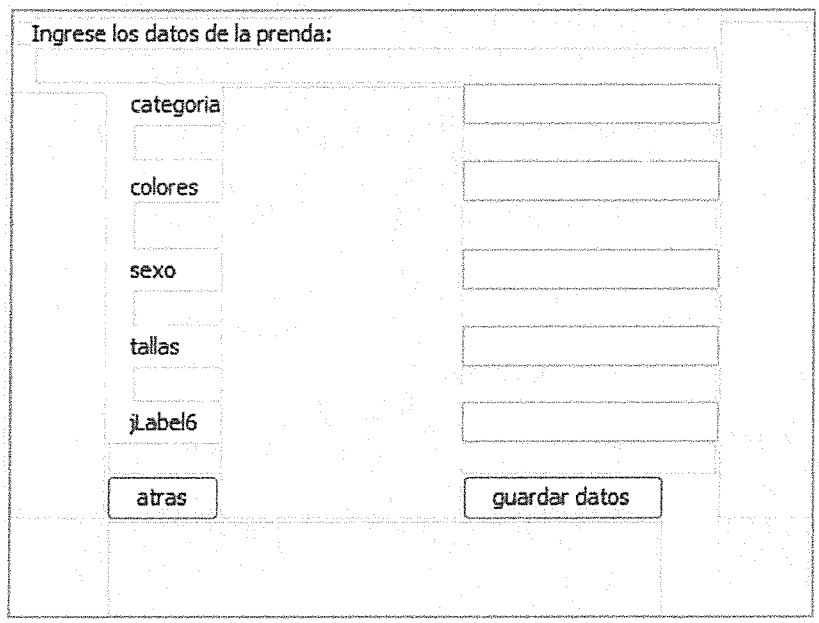
```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new registro().setVisible(true);
    this.hide();
  }

  private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
    // TODO add your handling code here:
    new busca().setVisible(true);
    this.hide();
  }

  private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
     new catalogo().setVisible(true);
    this.hide();
  }
```

- En esta parte del código se muestra la acción que tienen cada uno de los botones, esta acción es cambiar o mostrar la pantalla que representa a cada uno de ellos, es decir que si aplastamos es el botón para registrar la pantalla actual cambiara por la de registrar.
- Lo que realmente hace este código es abrir una nueva ventana según corresponda con el botón y este mismo sierra la ventana actual es decir esta que muestra las opciones, esa es la función de este código



(En la parte que se encuentra jlabel6 quiere decir costo)

Código:

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    DataOutputStream archivo=null;
    int clave=0;
    int tipo;
    int color;
    String talla;
    String costo;
    String sexo;

String nombre=new String("");

 BufferedReader teclado = new BufferedReader(new InputStreamReader(System.in));
```

```
try {

   archivo = new DataOutputStream( new FileOutputStream("prueba.txt",true) );

   clave=Integer.parseInt(jTextField2.getText());

   tipo=jTextField1.getText();
   color=Integer.parseInt(jTextField3.getText());
   talla=Integer.parseInt(jTextField4.getText());
   costo=""+iESS(tiempotrabajando, sueldo);
   sexo=""+despidoIn(tiempotrabajando, sueldo);
   categoria=""+indemnA(tiempotrabajando, sueldo);

   archivo.writeInt(clave);

   archivo.writeUTF(nombre);

   archivo.writeInt(tipo);

   archivo.writeInt(color);

   archivo.writeUTF(talla);
   archivo.writeUTF(sexo);
   archivo.writeUTF(costo);

   archivo.close();
  }
```

- En esta parte indica el registro de una nueva prenda o un nuevo articulo para esto el codigo que se muestra en forma previa sirve para hacer el guardado de los datos que se hayan colocado en la pantalla en forma respectiva este registro se hará en una carpeta fuera del programa dentro de la carpeta estará todo codificado para que la información se muestre como es realmente solo cuando se la muestra en el programa

Comparaciones entre los programas

Diferencias:

Programa 1 (programa base)	Programa 2 (programa modificado)
1.- En la parte de registro y el guardado solamente se puede registrar el código y el numero de las prendas.	En la parte de registro y guardado es posible registra el código, numero, colores y tipo de prenda que es.
2.- Para ingresar a ciertas funciones es necesario ingresar a otros lugares primero	Posee un acceso directo a cada función

Similitudes:

Programa 1 (programa base)	Programa 2 (programa modificado)
Su función es la misma	
Son hechos para varios usuarios secundarios y un principal	
Su sistema de búsqueda es el mismo	

Observaciones de la prueba:

Los comentarios de los usuarios del programa que lo probaron dieron comentarios positivos de la mejora diciendo que su uso es mas sencillo que el programa anterior, no solo se hizo probar a los empleados de la empresa, también se llamo a 10 personas las cuales no conocían ninguna de los dos programas, ha estas personas se les pidió que hicieran una serie de tareas en ambos programas después de hacerlo las 10 personas dijeron que era mas sencillo hacerlo por el programa numero 2, es decir el programa que se había modificado.

Conclucion de la monografía:

Como conclucion se puede decir que el mejoramiento del programa en su aspecto o forma visual si ayuda a una mejor y mas fácil forma de manipulación del programa, esto se puede decir ya que gracias a la prueba que se ha realizado esto el lo que demuestra la investigación.

Datos de lenguajes de programación:

"(Clasificación de lenguajes de programación:

Un lenguaje de programación es un lenguaje inventado para controlar una máquina, (normalmente, un ordenador). Hay muchísimos, de toda clase de tipos y características, inventados para facilitar el abordaje de distintos problemas, el mantenimiento del software, su reutilización, mejorar la productividad, etc.

Los lenguajes de programación se pueden clasificar según varios criterios. He encontrado **doce** en total: Nivel de abstracción, propósito, evolución histórica, manera de ejecutarse, manera de abordar la tarea a realizar, paradigma de programación, lugar de ejecución, concurrencia, interactividad, realización visual, determinismo y productividad.

Hay que tener en cuenta también, que en la práctica, la mayoría de lenguajes no pueden ser puramente clasificados en una categoría, pues surgen incorporando ideas de otros lenguajes y de otras filosofías de programación, pero no importa al establecer las clasificaciones, pues el auténtico objetivo de las mismas es mostrar los rangos, las posibilidades y tipos de lenguajes que hay.

1. Nivel de abstracción.

Según el nivel de abstracción, o sea, según el grado de cercanía a la máquina:

- Lenguajes de **bajo nivel**: La programación se realiza teniendo muy en cuenta las características del procesador. Ejemplo: Lenguajes ensamblador.
- Lenguajes de **nivel medio**: Permiten un mayor grado de abstracción pero al mismo tiempo mantienen algunas cualidades de los lenguajes de bajo nivel. Ejemplo: C puede realizar

operaciones lógicas y de desplazamiento con bits, tratar todos los tipos de datos como lo que son en realidad a bajo nivel (números), etc.
- Lenguajes de **alto nivel**: Más parecidos al lenguaje humano. Manejan conceptos, tipos de datos, etc., de una manera cercana al pensamiento humano ignorando (abstrayéndose) del funcionamiento de la máquina. Ejemplos: Java, Ruby.

Hay quien sólo considera lenguajes de bajo nivel y de alto nivel, (en ese caso, C es considerado de alto nivel).

2. Propósito.

Según el propósito, es decir, el tipo de problemas a tratar con ellos:

- Lenguajes de propósito **general**: Aptos para todo tipo de tareas: Ejemplo: C.
- Lenguajes de propósito **específico**: Hechos para un objetivo muy concreto. Ejemplo: Csound (para crear ficheros de audio).
- Lenguajes de **programación de sistemas**: Diseñados para realizar sistemas operativos o drivers. Ejemplo: C.
- Lenguajes de **script**: Para realizar tareas varias de control y auxiliares. Antiguamente eran los llamados lenguajes de procesamiento por lotes (batch) o JCL ("Job Control Languages"). Se subdividen en varias clases (de shell, de GUI, de programación web, etc.). Ejemplos: bash (shell), mIRC script, JavaScript (programación web).

3. Evolución histórica.

Con el paso del tiempo, se va incrementando el **nivel de abstracción**, pero en la práctica, los de una generación no terminan de sustituir a los de la anterior:

- Lenguajes de **primera generación (1GL)**: Código máquina.
- Lenguajes de **segunda generación (2GL)**: Lenguajes ensamblador.
- Lenguajes de **tercera generación (3GL)**: La mayoría de los lenguajes modernos, diseñados para facilitar la programación a los humanos. Ejemplos: C, Java.
- Lenguajes de **cuarta generación (4GL)**: Diseñados con un propósito concreto, o sea, para abordar un tipo concreto de problemas. Ejemplos: NATURAL, Mathematica.
- Lenguajes de **quinta generación (5GL)**: La intención es que el programador establezca el qué problema ha de ser resuelto y las condiciones a reunir, y la máquina lo resuelve. Se usan en inteligencia artificial. Ejemplo: Prolog.

4. Manera de ejecutarse.

Según la manera de ejecutarse:

- Lenguajes **compilados**: Un programa traductor traduce el código del programa (código fuente) en código máquina (código objeto). Otro programa, el enlazador, unirá los ficheros de código objeto del programa principal con los de las librerías para producir el programa ejecutable. Ejemplo: C.
- Lenguajes **interpretados**: Un programa (intérprete), ejecuta las instrucciones del programa de manera directa. Ejemplo: Lisp.

También los hay mixtos, como Java, que primero pasan por una fase de compilación en la que el código fuente se transforma en "bytecode", y este "bytecode" puede ser ejecutado luego (interpretado) en ordenadores con distintas arquitecturas (procesadores) que tengan todos instalados la misma "máquina virtual" Java.

5. Manera de abordar la tarea a realizar.

Según la manera de abordar la tarea a realizar, pueden ser:

- Lenguajes **imperativos**: Indican cómo hay que hacer la tarea, es decir, expresan los pasos a

realizar. Ejemplo: C.

- Lenguajes **declarativos**: Indican qué hay que hacer. Ejemplos: Lisp, Prolog. Otros ejemplos de lenguajes declarativos, pero que no son lenguajes de programación, son HTML (para describir páginas web) o SQL (para consultar bases de datos).

6. Paradigma de programación.

El **paradigma de programación** es el estilo de programación empleado. Algunos lenguajes soportan varios paradigmas, y otros sólo uno. Se puede decir que históricamente han ido apareciendo para facilitar la tarea de programar según el tipo de problema a abordar, o para facilitar el mantenimiento del software, o por otra cuestión similar, por lo que todos corresponden a lenguajes de alto nivel (o nivel medio), estando los lenguajes ensambladores "atados" a la arquitectura de su procesador correspondiente. Los principales son:

- Lenguajes de **programación procedural**: Divide el problema en partes más pequeñas, que serán realizadas por subprogramas (subrutinas, funciones, procedimientos), que se llaman unas a otras para ser ejecutadas. Ejemplos: C, Pascal.
- Lenguajes de **programación orientada a objetos**: Crean un sistema de clases y objetos siguiendo el ejemplo del mundo real, en el que unos objetos realizan acciones y se comunican con otros objetos. Ejemplos: C++, Java.
- Lenguajes de **programación funcional**: La tarea se realiza evaluando funciones, (como en Matemáticas), de manera recursiva. Ejemplo: Lisp.
- Lenguajes de **programación lógica**: La tarea a realizar se expresa empleando lógica formal matemática. Expresa qué computar. Ejemplo: Prolog.

Hay muchos paradigmas de programación: Programación genérica, programación reflexiva, programación orientada a procesos, etc.

7. Lugar de ejecución.

En **sistemas distribuidos**, según dónde se ejecute:

- Lenguajes de **servidor**: Se ejecutan en el servidor. Ejemplo: PHP es el más utilizado en servidores web.
- Lenguajes de **cliente**: Se ejecutan en el cliente. Ejemplo: JavaScript en navegadores web.

8. Concurrencia.

Según admitan o no concurrencia de procesos, esto es, la ejecución simultánea de varios procesos lanzados por el programa:

- Lenguajes **concurrentes**. Ejemplo: Ada.
- Lenguajes **no concurrentes**. Ejemplo: C.

9. Interactividad.

Según la interactividad del programa con el usuario u otros programas:

- Lenguajes **orientados a sucesos**: El flujo del programa es controlado por la interacción con el usuario o por mensajes de otros programas/sistema operativo, como editores de texto, interfaces gráficos de usuario (GUI) o kernels. Ejemplo: VisualBasic, lenguajes de programación declarativos.
- Lenguajes **no orientados a sucesos**: El flujo del programa no depende de sucesos exteriores, sino que se conoce de antemano, siendo los procesos batch el ejemplo más claro (actualizaciones de bases de datos, colas de impresión de documentos, etc.). Ejemplos: Lenguajes de programación imperativos.

10. Realización visual.

Según la realización visual o no del programa:

- Lenguajes de **programación visual**: El programa se realiza moviendo bloques de construcción de programas (objetos visuales) en un interfaz adecuado para ello. No confundir con entornos de programación visual, como Microsoft Visual Studio y sus lenguajes de programación textuales (como Visual C#). Ejemplo: Mindscript.
- Lenguajes de **programación textual**: El código del programa se realiza escribiéndolo. Ejemplos: C, Java, Lisp.

11. Determinismo.

Según se pueda predecir o no el siguiente estado del programa a partir del estado actual:

- Lenguajes **deterministas**. Ejemplos: Todos los anteriores.
- Lenguajes **probabilísticos o no deterministas:** Sirven para explorar grandes espacios de búsqueda, (como gramáticas), y en la investigación teórica de hipercomputación. Ejemplo: mutt (generador de texto aleatorio).

12. Productividad.

Según se caractericen por tener virtudes útiles o productivas, u oscuras y enrevesadas:

- Lenguajes **útiles o productivos:** Sus virtudes en cuanto a eficiencia, sencillez, claridad, productividad, etc., motiva que sean utilizados en empresas, administraciones públicas y/o en la enseñanza. Ejemplos: Cualquier lenguaje de uso habitual (C, Java, C++, Lisp, Python, Ruby, ...).
- Lenguajes **esotéricos o exóticos:** Inventados con la intención de ser los más raros, oscuros, difíciles, simples y/o retorcidos de los lenguajes, para diversión y entretenimiento de frikis programadores. A veces exploran nuevas ideas en programación. Ejemplo: Brainfuck.)"

(todo lo que se a puesto desde Clasificación de lenguajes de programación, se lo a sacado de la siguiente pagina:

http://qbitacora.wordpress.com/2007/09/21/clasificacion-de-lenguajes-de-programacion/)

Bibliografía:

http://qbitacora.wordpress.com/2007/09/21/clasificacion-de-lenguajes-de-programacion/

Numero total de palabras: 3028