ESCUELA SUPERIOR DE INGENIERÍA

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

Rules. Lenguaje para el desarrollo de software dentro del ámbito de los juegos de mesa

Fco. Javier Bohórquez Ogalla

25 de marzo de 2015

ESCUELA SUPERIOR DE INGENIERÍA

GRADO EN INGENIERÍA EN INFORMÁTICA

PLANTILLA PARA PROYECTO DE FIN DE GRADO

- Departamento: Lenguajes y Sistemas informáticos
- Director del proyecto: Profesor Profesor Profesor
- Autor del proyecto: Alumno Alumno Alumno

Cádiz, 25 de marzo de 2015

Fdo: Alumno Alumno Alumno

Agradecimientos

Introduzca aquí, si lo desea, los agradecimientos.

Resumen

Introduzca aquí un resumen no superior a 500 palabras, que servirá de descripción pública del trabajo realizado.

Palabras clave: Lista de palabras clave que reflejen el contenido del trabajo en aras de facilitar su búsqueda en sistemas bibliográficos.

Índice general

Índice de figuras

Índice de cuadros

Parte I

Prolegómeno

La primera parte de la memoria del PFC debe contener una introducción y una planificación del proyecto.

La introducción es un capítulo que, a modo de resumen, debe contener una breve descripción del contexto de la disciplina en la que el proyecto tiene aplicación y la motivación para su desarrollo, así como del alcance previsto.

El segundo capítulo debe incluir una planificación del proyecto. La planificación deberá ajustarse a las prácticas de ingeniería en general, y de la ingeniería del software en particular. Deberá tener en cuenta los plazos, los entregables (documentos y software), los recursos (humanos y de equipamiento inventariable) y el método de ingeniería de software a emplear.

Capítulo 1

Introducción

A continuación, se describe la motivación del presente proyecto y su alcance. También se incluye un glosario de términos y la organización del resto de la presente documentación.

1.1. Motivación

Qué motivación nos ha llevado a su desarrollo. Contexto y ámbito en el que se desarrolla el proyecto.

1.2. Alcance

Esta sección debe describir a qué elementos organizativos de la organización Cliente afecta el desarrollo del nuevo sistema. También debe describir los principales objetivos que se esperan alcanzar cuando el sistema a desarrollar esté en producción.

1.3. Glosario de Términos

Esta sección debe contener una lista ordenada alfabéticamente de los principales términos, acrónimos y abreviaturas específicos del dominio del problema, especialmente de los que se considere que su significado deba ser aclarado. Cada término, acrónimo o abreviatura deberá acompañarse de su definición.

1.4. Organización del documento

Descripción de los contenidos de la presente memoria, así como del software entregado en soporte informático.

Capítulo 2

Planificación

En esta sección se describen todos los aspectos relativos a la gestión del proyecto: metodología, organización, costes, planificación, riesgos y aseguramiento de la calidad.

2.1. Metodología de desarrollo

Definición del proceso de desarrollo, ciclo de vida y metodología empleada durante la elaboración del proyecto. Las fases y/o iteraciones que proponga el método empleado deberán quedar recogidas en la planificación que se detalle más adelante.

2.2. Planificación del proyecto

Estimación temporal y definición del calendario básico (hitos principales e iteraciones). Desarrollo de la planificación detallada, utilizando un diagrama de Gantt. Los diagramas de Gantt que se vean correctamente (girados y divididos si hace falta).

Se debe incluir una comparación cuantitativa del tiempo y el esfuerzo realmente invertido frente al estimado y planificado. Estos datos pueden recogerse del sistema de gestión de tareas empleado para el seguimiento del proyecto.

2.3. Organización

Relación de las personas (roles) involucradas en el proyecto y de cómo se estructuran las relaciones entre las mismas para ejecutar el proyecto. Relación de los recursos inventariables utilizados en el proyecto: equipamiento informático (hardware y software), herramientas empleadas, etc.

2.4. Costes

Estudio y presupuesto de los costes de los recursos (humanos y materiales) descritos anteriormente, necesarios para el proyecto.
Para el cálculo de costes de personal pueden consultarse las tablas salariales de la UCA para el personal técnico de apoyo contratado laboral [**?**], o bien otras más ajustadas a la realidad. El cálculo del coste del personal del proyecto debe hacerse en personas-mes, y luego hacer la

correspondencia al coste monetario.

2.5. Riesgos

Enumeración de los riesgos del proyecto, indicando su posible impacto (efecto que la ocurrencia del citado riesgo tendría en el desarrollo del proyecto) y la probabilidad de ocurrencia. Una vez los riesgos son identificados y priorizados, hay que definir los planes necesarios para reducir los efectos del riesgo una vez se haya materializado o disminuir que este ocurra.

2.6. Aseguramiento de calidad

En esta sección se incluirán las actividades y tareas relacionadas con el aseguramiento de calidad a realizar durante el desarrollo del software. Se incluirán los estándares, prácticas y normas aplicables durante el desarrollo del software.

También, deberán recogerse los diferentes tipos de revisiones, verificaciones y validaciones que se van a llevar a cabo, los criterios para la aceptación o rechazo de cada producto y los procedimientos para implementar acciones correctoras o preventivas.

Parte II

Desarrollo

En esta parte se debe describir el desarrollo del proyecto siguiendo la metodología empleada. Sus capítulos no deben ser una descripción exhaustiva de todos los documentos, diagramas, código fuente y, en general, entregables generados, sino más bien una explicación resumida del desarrollo, estructurada según las etapas principales del proceso de ingeniería. Deben seleccionarse aquellos diagramas, fragmentos de código y secciones de los entregables que sean más significativos para dicha explicación. La totalidad de los entregables resultado del proyecto se ubicarán en los anexos y/o en el material en CD/DVD que acompañe al proyecto.

Capítulo 3

Requisitos del Sistema

En esta sección se detalla la situación actual de la organización y las necesidades de la misma, que originan el desarrollo o mejora de un sistema informático. Luego se presentan los objetivos y el catálogo de requisitos del nuevo sistema. Finalmente se describen las diferentes alternativas tecnológicas y el análisis de la brecha entre los requisitos planteados y la solución base seleccionada, si aplica.

3.1. Situación actual

Esta sección debe contener información sobre la situación actual de la organización para la que se va a desarrollar el sistema software.

3.1.1. Procesos de Negocio

Esta sección debe contener información sobre los modelos de procesos de negocio actuales, que suelen ser la base de los modelos de procesos de negocio a implantar.

3.1.2. Entorno Tecnológico

Esta sección debe contener información general sobre el entorno tecnológico en la organización del cliente antes del comienzo del desarrollo del sistema software, incluyendo hardware, redes, software, etc.

3.1.3. Fortalezas y Debilidades

Esta sección debe contener información sobre los aspectos positivos y negativos del negocio actual de la organización para la que se va a desarrollar el sistema software.

3.2. Necesidades de Negocio

Esta sección debe contener información sobre los objetivos de negocio de clientes y usuarios, incluyendo los modelos de procesos de negocio a implantar.

3.2.1. Objetivos de Negocio

Esta sección debe contener los objetivos de negocio que se esperan alcanzar cuando el sistema software a desarrollar esté en producción.

3.2.2. Procesos de Negocio

Esta sección, debe contener los modelos de procesos de negocio a implantar, que normalmente son los modelos de procesos de negocio actuales con ciertas mejoras.

3.3. Objetivos del Sistema

Esta sección debe contener la especificación de los objetivos o requisitos generales del sistema.

3.4. Catálogo de Requisitos

Esta sección debe contener la descripción del conjunto de requisitos específicos del sistema a desarrollar para satisfacer las necesidades de negocio del cliente.

3.4.1. Requisitos funcionales

En esta subsección se detallan los requisitos funcionales del sistema. Estos se han organizado en distintas categorías, según cuestiones realativas al software como intérprete y según las distintas características del lenguaje.
La especificación de requerimientos funcionales que se presenta es completa y consistente, dado que recoje todos los servicios y necesidades que se pretenden cubrir y evita presentar ambiguedades o definiciones contradictorias.

.

Número: 0001

Nombre: Léxico.

Categoría: Interprete.

Descripción: El sistema debe fijar el léxico del lenguaje conformado por una conjunto de palabras y expresiones bien definidas y acotadas.

Número: 0002

Nombre: Gramática.

Categoría: Interprete.

Descripción: El sistema debe definir una gramática que representará el lenguaje. La gramática debe ser libre de contexto, clara y uniforme en toda su extensión. Además debe estar libre de ambigüedades.

Número: 0003

Nombre: Comentarios.

Categoría: Interprete.

Descripción: Se ha de contemplar un mecanismo para añadir comentarios al contenido fuente que serán ignorados durante la tarea de interpretación.

Los comentarios comprenderán desde un carácter "#", o bien "", hasta fin de línea.

Por otro lado se ha de contemplar los comentarios de múltiples líneas, que deerán estar contenidos entre "" y "".

Número: 0004

Nombre: Interpretación semántica.

Categoría: Interprete.

Descripción: Dado un contenido fuente el sistema debe analizarlo en función al léxico (análisis léxico) y la gramática (análisis sintáctico) del lenguaje y producir el resultado semántico asociado.

Número: 0005

Nombre: Fuente desde línea de comandos.

Categoría: Interprete.

Descripción: El interprete debe ser capaz de obtener contenido fuente desde una línea de comandos.

Número: 0006

Nombre: Fuente desde entrada estándar.

Categoría: Interprete.

Descripción: El interprete debe ser capaz de obtener contenido fuente desde la entrada estándar del sistema.

Número: 0007

Nombre: Fuente desde fichero.

Categoría: Interprete.

Descripción: El interprete debe ser capaz de obtener contenido fuente desde un fichero.

Número: 0008

Nombre: Entorno de ejecución.

Categoría: Interprete.

Descripción: El interprete debe definir un entorno de ejecución en el que se controlen parámetros de entrada, variables de entornos del sistema operativo e información sobre el proceso como número de línea actual y los errores producidos.

Número: 0009

Nombre: Información de errores.

Categoría: Interprete.

Descripción: Si se produce un error el interprete debe informar del tipo y causa del error, así como del contexto en el que se ha producido (nombre y línea de fichero).

Número: 0010

Nombre: Sentencia.

Categoría: Interprete.

Descripción: Son las unidades interpretables más pequeña en las que se divide un contenido fuente. Las sentencias están sujetas a unas reglas sintácticas y encierran un significado semántico. El interprete debe definir la gramática de cada sentencia y dotarlas de significado semántico. Toda sentencia debe finalizar con el carácter ";", excluyendo las sentencias formadas por bloques de sentencias. Aunque carezca de sentido práctico, para evitar posibles errores de codificación y mantener coherencia en la sintaxis y la definición del lenguaje, se debe contemplar la sentencia vacía que sólo conste del carácter ";".

Número: 0011

Nombre: Bloques de sentencias.

Categoría: Interprete.

Descripción: Son un conjunto de sentencias que deberán ser interpretadas y ejecutadas secuencialmente. La disposición de sentencias en el bloque determinan el flujo de ejecución que se llevará a cabo cuando se interprete el bloque. El contenido fuente en si mismo es un bloque de sentencias. Todo bloque de sentencias de más de una sentencia (con excepción del contenido fuente en si mismo) debe ir delimitado mediante llaves. Aunque no sea de uso común, para mantener coherencia en la sintaxis y la definición del lenguaje, se debe contemplar el bloque de sentencias vacío.

Número: 0012

Nombre: Elementos imprimibles.

Categoría: Interprete.

Descripción: El interprete debe definir y operar sobre una serie de elementos imprimibles, estos pueden ser datos, expresiones, sentencias u otros elementos del lenguaje. Un elemento imprible debe tener una representación gráfica que permita mostrarlo en pantalla.

Nombre: Datos.

Categoría: Interprete.

Número: 0013

Descripción: El interprete deberá operar sobre datos. El contenido fuente definirá cómo se han de construir y/o acceder a los datos y las operaciones que se realizarán sobre ellos durante la ejecución.

Un dato será tratado en función un tipo de dato. El tipo de dato lo dota de una semántica, un significado. Así, todo dato deberá ser considerado un objeto, por lo que tendrán unas propiedades y funcionalidad ligadas al tipo como el que es trarado.

Número: 0014

Nombre: Expresiones.

Categoría: Interprete.

Descripción: El interprete debe ser capaz de evaluar expresiones. Estas son secuencias de datos, operadores, operandos, elementos de puntuación y/o palabras clave, que especifican una unidad computacional. Generalmente encierran un valor que se asocia a la expresión después de ser evaluada. Una sentencia puede estar formada por una o varias expresiones que deberán ser evaluadas o interpretadas para dotarla de significado. Una sentencia puede constar únicamente de una expresión en ese caso la sentencia es considerada la evaluación de dicha expresión.

La expresión más simple equivale a un único dato, en este caso el valor de la expresión será el del dato.

Número: 0015

Nombre: Expresiones de tipo definido.

Categoría: Interprete.

Descripción: Son expresiones cuyo valor es de un tipo definido y fijo. El sistema debe interpretar estas expresiones para determinar el valor asociado a la mismas en un momento dado.

Nombre: Expresiones de tipo no definido.

Categoría: Interprete.

Número: 0016

Descripción: Son expresiones cuyo valor no tiene un tipo definido ni fijo, sino que es durante la interpretación cuando se determina el tipo. El sistema debe interpretar estas expresiones para determinar, además del valor asociado a la mismas, el tipo de dato que guardan en un momento dado.

Nombre: Listado de expresiones.

Categoría: Interprete.

Número: 0017

Descripción: Se ha de facilitar un mecanismo que permita agrupar expresiones para darles un significado operacional común. Este deberá consistir en una serie de expresiones separadas por comas.

Nombre: Pares de expresiones.

Categoría: Interprete.

Número: 0018

Descripción: Se ha de facilitar un mecanismo que permita relacionar un par de expresiones para darles un significado estructural. Este deberá consistir en el par de expresiones separadas por el carácter ":".

Número: 0019

Nombre: Tipos de datos.

Categoría: Interprete.

Descripción: El sistema debe ser capaz de interpretar y operar sobre diferentes tipos de datos. Las expresiones pueden tener un tipo de dato asociado que puede o no ser definido y fijo. Los tipos de datos pueden ser simples o compuestos.

Un dato debe tratarse como diferente tipo en función el contexto en el que se utilice. Así un dato de un tipo concreto puede ser tratado como otro tipo de dato si fuese necesario. Un dato por si mismo siempre será considerado del tipo de dato con el que se creó, sin embargo cuando interviene en una operación es posible que se precise una conversión o equivalencia de tipos. Para ello debe tomar su valor como si de otro tipo se tratase. Si en la operación no es posible convertir el tipo en el tipo requerido se debe producir un error de tipos.

Se debe establecer un mecanismo de conversión de tipos. La relación de conversión de tipo debe ser transitiva, así si un dato de tipo lógico puede verse como un dato de tipo aritmético y un dato aritmético como un cadena de caracteres, entonces el dato de tipo lógico también puede verse como una cadena.

Todo tipo de dato representa en s

Número: 0020

Nombre: Tipo de dato nulo.

Categoría: Tipo de dato simple.

Descripción: Se debe contemplar el tipo de dato nulo. Este tipo de dato tendrá un único valor posible. El valor nulo deberá representar un elemento no definido. Una expresión puede tomar el valor nulo cuando sea evaluada si se hace uso de elementos no definidos.

Número: 0021

Nombre: Tipo de dato lógico.

Categoría: Tipo de dato simple.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo lógicos. Este tipo de dato sólo contempla dos posibles valores: falso y verdadero. Este será el tipo de dato más simple. Un dato lógico puede ser tratado como un tipo de dato aritmético tomándose falso como el valor cero, y verdadero como el valor uno. Los datos de tipo lógico deben ser elementos imprimibles.

Número: 0022

Nombre: Tipo de dato aritmético.

Categoría: Tipo de dato simple.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo aritméticos. Este tipo de dato contempla valores numéricos racionales. Todo dato aritmético además tiene asociado un valor lógico cuando se utiliza como este tipo de dato, tal que cualquier número distinto de cero tiene valor verdadero y el cero tiene el valor falso. Además cuando un dato aritmético es tratado como una cadena de caracteres se tomará la cadena que representa al número. Los datos de tipo aritmético deben ser elementos imprimibles.

Número: 0023

Nombre: Tipo de dato cadenas de caracteres

Categoría: Tipo de dato compuesto

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo cadena de caracteres. Este tipo de dato contempla cualquier sucesión de caracteres alfanuméricos, secuencias de escape, u otros signos o símbolos. Esta sucesión pude ser vacía. Una cadena de caracteres vendrá delimitada mediante comillas dobles o simples. Toda cadena de caracteres además tiene asociado un valor aritmético cuando se utiliza como este tipo de dato, tal que, si la cadena representa un número racional el valor será el del propio número, por otro lado, si la cadena no representa un número racional el valor aritmético de la misma será el número de caracteres que la conforman. Los datos de tipo cadena de caracteres deben ser elementos imprimibles. No se ha de considerar el tipo simple de dato carácter, pudiéndose tratar este como una cadena de un solo elemento.

Número: 0024

Nombre: Tipo de dato expresión regular.

Categoría: Tipo de dato compuesto.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo expresión regular. Una expresión regular consiste en una cadena de caracteres que representan un patrón. Las expresiones regulares tendrán una sintaxis PERL. Una expresión regular se delimita mediante caracteres acento grave (`). El tipo de dato expresión regular no debe ser tratado como otro tipo de dato.

Número: 0025

Nombre: Tipo de dato array.

Categoría: Tipo de dato compuesto.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo array. Este tipo de dato contempla cualquier sucesión de elementos. Estos elementos pueden ser pares de expresiones clave/valor donde la clave servirá para referenciar el valor dentro de la sucesión. También pueden ser simples expresiones por lo que se tomará automáticamente una clave numérica y secuencial según el orden del array y como valor el de la expresión. El significado semántico de las claves en un array puede ser numérico (array numérico) o cadenas de caracteres (array asociativo). Los elementos del array serán ordenados por defecto de forma alfabética en función a la clave. Una definición de array se delimita mediante llaves y sus elementos se denotarán mediante un listado de expresiones o pares de estas. Los datos de tipo array deben ser elementos imprimibles. Un dato de tipo array solo puede ser tratado como un tipo de dato booleano, siendo falso si se encuentra vacío y verdadero en caso contrario.

Número: 0026

Nombre: Tipo de dato etiqueta.

Categoría: Tipo de dato funcional.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo etiqueta. Una etiqueta es una referencia a una sentencia concreta dentro del contenido fuente. Las etiquetas dependen quedarán definidas dentro de un contexto determinado por el bloque de sentencias en el que se encuentren.

Número: 0027

Nombre: Tipo de dato función.

Categoría: Tipo de dato funcional.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo funciones. Las funciones se definen mediante una serie de parámetros de entradas y un bloque de sentencias. Tras ser interpretada la función toma un valor que puede ser de cualquiera de los tipos de datos soportados.

Número: 0028

Nombre: Tipo de dato objeto.

Categoría: Tipo de dato funcional.

Descripción: El sistema debe ser capaz de interpretar y operar sobre datos de tipo objeto. Los objetos serán vistos como estructuras datos funcionales, referenciando tanto datos de diferentes tipos (atributos), como funciones (métodos). Un objeto puede o no pertenecer a una clase de objetos. Estos, los objetos, deberán ser tratados como un tipo especial de array donde ciertos valores son bloques de sentencias (funciones). Dentro del bloque de sentencias que conforma un método es posible hacer referencia a los demás valores del objeto mediante la expresión especial "this". Los datos de tipo objeto deben ser elementos imprimibles, además los objetos deberían poder definir su propio método de impresión. Al igual que un array podrán ser tratados como tipos de datos lógicos, pero además, si definen un método de impresión podrán ser tratados como cadenas de caracteres cuyo valor será la cadena devuelta por el método.

Número: 0029

Nombre: Tipo de dato clase de objeto.

Categoría: Tipo de dato funcional.

Descripción: El lenguaje debe contemplar el paradigma de la programación orientada a objetos. Una clase se ha de ver como una definición estática e inmutable que será utilizada para la creación de objetos. Las clases deberán ser vistas como un tipo de datos, siendo representadas por objetos prototipos que serán clonados e inicializados. El usuario podrá crear objetos a partir de una clase ya definida, mediante la clonación de otro objeto, o directamente mediante sentencias y expresiones. Las clases de objetos pueden definir un método constructor que será utilizado cuando se cree un objeto a partir de la misma. Las clases de objetos deben poder relacionarse entre si por una relación de herencia, tomando la clase hija la definición y valores de la padre.

Número: 0030

Nombre: Identificadores.

Categoría: Tabla de símbolos.

Descripción: El interprete debe facilitar mecanismos para que el usuario defina e identifique expresiones, datos, bloques de sentencias, y otras construcciones y elementos del lenguaje. Se precisa una manera unívoca de nombrar estos elementos. Un identificador válido debe estar formado por una secuencia de caracteres alfanuméricos de al menos un carácter, donde el primer carácter a de ser una letra.

Número: 0031

Nombre: Tabla de símbolos.

Categoría: Tabla de símbolos.

Descripción: El interprete debe ser capaz de gestionar tablas de símbolos. Los símbolos hacen referencias a valores, funciones y otras expresiones del lenguajes. Para acceder a estos símbolos se debe utilizar un identificador. Se hace necesario el acceso y uso de los símbolos según el contexto de ejecución, determinado por el ámbito y el tipo símbolo, para ello deben poder coexistir diferentes tablas de símbolos globales. Para evitar conflictos en el uso de identificadores algunos tipos de datos compuestos deben disponer de su propia tabla de símbolos.

Número: 0032

Nombre: Símbolos variables.

Categoría: Tabla de símbolos.

Descripción: El interprete debe ser capaz gestionar una serie de símbolos denominados variables. Estos relacionan un identificador con un valor que puede variar durante el proceso de ejecución. El tipo de una variable dependerá del tipo del valor al que referencia (tipado dinámico), este podría ser de cualquiera de los tipos de datos soportados. La tabla de símbolos de variables debe adaptarse al contexto de ejecución.

Número: 0033

Nombre: Variables globales.

Categoría: Tabla de símbolos.

Descripción: Aunque la tabla de símbolos de variables es dependiente del contexto de ejecución se ha de facilitar algún mecanismo para que un dato esté disponible independientemente del contexto en el que se acceda.

Número: 0034

Nombre: Símbolos funciones.

Categoría: Tabla de símbolos.

Descripción: El interprete debe ser capaz gestionar una serie de símbolos denominados funciones. Aunque un dato de tipo función puede ser almacenado en una variable, se debe facilitar una tabla de símbolos especial para funciones, en la cual el programador podrá referenciar solo este tipo de datos. El objetivo de esta tabla es poder utilizar identificadores para funciones sin causar posibles conflictos con otros identificares. La tabla de símbolos de funciones es global e independiente del contexto.

Número: 0035

Nombre: Símbolos clases.

Categoría: Tabla de símbolos.

Descripción: El interprete debe ser capaz gestionar una serie de símbolos denominados clases. A diferencia de una función, una clase no puede almacenarse en una variable debido a la naturaleza inmutable de la misma. Se ha de facilitar una estructura para almacenar y referenciar las clases definidas por el usuario. La tabla de símbolos de clases es global e independiente del contexto.

Número: 0036

Nombre: Sentencia typeof.

Categoría: Sentencia.

Descripción: Debido al tipado dinámico se precisa de un mecanismo para conocer el tipo de dato relacionado con una variable. Este deberá mostrar en la salida estándar el tipo de la variable asociada a un identificador dado.

Número: 0037

Nombre: Sentencia de salida de datos.

Categoría: Sentencia.

Descripción: Se debe disponer de una sentencia que permita llevar acabo la salida de datos. Estas supondrán un mecanismo para que el contenido fuente pueda mostrar en la salida estándar los datos sobre los que opera. Estos datos debes ser elementos imprimibles.

Número: 0038

Nombre: Sentencia de entrada de datos.

Categoría: Sentencia.

Descripción: El interprete debe implementar algún recurso que permita que el contenido fuente del usuario obtenga datos de la entrada estándar para operar. Este mecanismo deberá dar la posibilidad de mostrar un prompt. Además debe dar la opción de que la entrada finalice al introducir un salto de línea o al generarse una señal EOF (end-of-file).

Número: 0039

Nombre: Sentencia de finalización de ejecución.

Categoría: Sentencia.

Descripción: Se ha de disponer de un mecanismo para que el sistema finalice de forma inmediata de interpretar el contenido fuente.

Número: 0040

Nombre: Sentencia sleep.

Categoría: Sentencia.

Descripción: Se debera de proporcionar un mecanismo que permita suspender o bloquear la ejecución durante un tiempo dado. Constará de una expresión que represente el valor aritmético del tiempo en segundos.

Número: 0041

Nombre: Sentencia include.

Categoría: Sentencia de control de flujo.

Descripción: Se ha de facilitar un mecanismo para incluir en un punto de la ejecución contenido fuente localizado en recurso externo. El recurso consistirá en un fichero con sentencias interpretables.

Número: 0042

Nombre: Sentencia goto.

Categoría: Sentencia de control de flujo.

Descripción: Se ha de facilitar un mecanismo para llevar el flujo de ejecución a la sentencia referenciada por una etiqueta.

Número: 0043

Nombre: Sentencia if.

Categoría: Sentencia de control de flujo.

Descripción: Deben de existir una serie de sentencias condicionales que alteren el flujo de ejecución. Las sentencias if deberán estar construidas por bloques de sentencias y una serie de expresiones denominadas "condiciones". La interpretación de una sentencia de este tipo debe consistir en la evaluación lógica de las "condiciones" para determinar el bloque de sentencias que se ejecutará. Las formas de la sentencia if que el interprete debe aceptar son las siguientes:

- if (cond) stmts
- if (cond) stmts else stmts
- if (cond) stmts elif (cond) stmts ... else stmts

Número: 0044

Nombre: Sentencia switch case.

Categoría: Sentencia de control de flujo.

Descripción: El interprete debe ser capaz de interpretar sentencias del tipo switch case. Estas constan de una lista de bloques de sentencias precedidas de una expresión denominada "caso". Dada una expresión base esta debe ser comparada mediante la operación de igualdad con cada uno de los casos, ejecutando el bloque correspondiente al "caso" cuya comparación sea positiva y todos los bloques siguientes. Se deberá poder especificar un bloque denominado "default" que no dispondrá de expresión "caso" y será ejecutado sin aplicar condición alguna.

Número: 0045

Nombre: Sentencia while.

Categoría: Sentencia de control de flujo.

Descripción: El interprete debe ser capaz de interpretar sentencias del tipo while. Esta es una sentencia de control iterativa que consta de una expresión denominada "condición" y un bloque de sentencias. El bloque de sentencias debe ser ejecutado mientras que "condición" permanezca verdadera.

Número: 0046

Nombre: Sentencia do while.

Categoría: Sentencia de control de flujo.

Descripción: El interprete debe ser capaz de interpretar sentencias del tipo do while. Esta es una sentencia de control iterativa que consta de una expresión denominada "condición" y un bloque de sentencias. El bloque de sentencias debe ser ejecutado mientras que "condición" permanezca verdadera, llevándose a cabo la ejecución al menos una vez.

Número: 0047

Nombre: Sentencia for.

Categoría: Sentencia de control de flujo.

Descripción: El interprete debe ser capaz de interpretar sentencias del tipo for. Esta es una sentencia de control iterativa que consta de tres expresiones denominadas "inicialización", "condición" y "paso", además de un bloque de sentencias. Primero se ha de evaluar la expresión "inicialización", luego el bloque de sentencias se ejecutará mientras "condición" se valore como verdadera. La expresión "paso" se deberá ejecutar al finalizar cada iteración.

Número: 0048

Nombre: Sentencia break.

Categoría: Sentencia de control de flujo.

Descripción: Se ha de disponer de un mecanismo para indicar que el flujo debe salir de una sentencia de control. Se ha de contemplar las sentencias anidadas.

Número: 0049

Nombre: Sentencia foreach.

Categoría: Sentencia de control de flujo.

Descripción: Se han de de interpretar sentencias del tipo forearch. Esta es una sentencia de control iterativa que consta un bloque de sentencias, de una expresión denominada "conjunto" y un identificador denominado "valor". Se debe poder, aunque de forma opcional, especificar otro identificador que se denominará "clave". El bloque de sentencias será ejecutado de forma iterativa en función el tipo de dato y valor de "conjunto". El "conjunto" será evaluado para determinar el número de iteraciones y el valor que se le asignará como variables a los identificadores en cada iteración. Dependiendo del tipo de la expresión "conjunto" la sentencia foreach deberá actuar como sigue:

Tipo lógico: El bloque de sentencias se ejecutará mientras "conjunto" sea verdadero. El identificador "valor" tomará el valor verdadero. En el caso en el que se especifique un identificador "clave" a este no se le asignará ningún valor.

Tipo aritmético: Si "conjunto" representa un número mayor que cero el bloque de sentencias se ejecutará tantas veces como el valor numérico que representa. En cada iteración "valor" se le asignará el número de la iteración comenzando por cero. Si se presenta un identificador "clave" a este no se le asignará ningún valor. Si el valor de "conjunto" es menor o igual a cero el bloque no deberá ejecutarse.

Tipo cadena de caracteres: Si "conjunto" es una cadena de caracteres que representa un número racional la ejecución deberá ser como si de un tipo aritmético se tratase. Si el "conjunto" es una cadena que no representa un número racional el bloque de sentencias se ejecutará por cada carácter en la cadena. En este último caso a "valor" se le asignará el carácter contemplado en cada iteración. Si se dio un identificador "clave" este no será asignado.

Tipo array: Si "conjunto" es un array, u otro tipo de dato derivado de este como un objeto, el bloque de sentencias se ejecuta por cada elemento en el mismo. Al identificador "valor" se le asignará el valor del elemento en el array correspondiente a la iteración. En el caso de que se facilite un identificador "clave" este deberá tomar la clave del elemento en el array.

Otros tipos: No se llevará a cabo ninguna operación.

Número: 0050

Nombre: Sentencia continue.

Categoría: Sentencia de control de flujo.

Descripción: El sistema debe facilitar algún recurso que permita finalizar la iteración actual de una sentencia de control en ejecución y comenzar con la siguiente. Este mecanismo debe contemplar la posibilidad de salir de varias sentencias de control anidadas.

Número: 0051

Nombre: Sentencia ciclo incremental.

Categoría: Sentencia de control de flujo.

Descripción: Se ha de facilitar una sentencia de control que permita iterar un bloque de sentencias en función a un rango de números enteros con la forma $[0 - n]$. Cada valor del rango en cada iteración deberá asignarse a un identificador como variable, de forma que sea accesible desde el bloque de sentencias.

Número: 0052

Nombre: Sentencia ciclo ágil.

Categoría: Sentencia de control de flujo.

Descripción: Se ha de facilitar una sentencia de control que permita iterar un bloque de sentencias en función una expresión "conjunto" de forma ágil y sencilla. Para ello esta sentencia deberá operar igual que la sentencia foreach pero sin ser necesario, aunque posible, dar un identificador "valor" sobre el que se realizará la asignación. En lugar de ello la asignación que se produce en cada iteración se deberá realizar sobre un símbolo con identificador fijo y contenido variable denominado iterador. El iterador debe ser accesible desde el bloque de sentencias. Además se debe contemplar el acceso al iterador de varias sentencias de ciclo ágil cuando estas se presentan de forma anidada.

Número: 0053

Nombre: Operadores.

Categoría: Operadores.

Descripción: Se ha de facilitar una serie de operadores que permitan manipular los datos. Los operadores son en si mismo expresiones, por los que estos tendrán un valor asociado tras ejecutarse. Los operadores constarán de una serie operandos que intervendrán en la operación y que serán a su vez otras expresiones. Los operadores se clasificarán en función del tipo de valor que tendrán tras la ejecución, los tipos de los operandos y/o la naturaleza del operador en si.

Número: 0054

Nombre: Acceso a símbolo variable.

Categoría: Operadores sobre símbolos.

Descripción: Se hace necesario la gestión de los símbolos variables creados durante la ejecución, lo que implica el acceso a los datos referenciados por estos. Se deberá poder acceder a los datos referenciados por una variable simplemente mediante una expresión formada por el identificador asociado a la misma. El acceso a una variable debe originar un valor por defecto si esta no tiene un valor asociado. El valor por defecto dependerá del contexto:

Clave de array: Si la variable se utiliza para la clave de un array, o cualquier tipo derivado de este, el valor por defecto de esta es una cadena de caracteres que representa el identificador con el que se accede a la misma.

Cualquier otro: En cualquier otro contexto el valor por defecto será el valor nulo.

Una operación de acceso a variable representa una expresión que deberá tener, como valor asociado tras su ejecución, el valor al que referencia el símbolo variable.

Número: 0055

Nombre: Asignación.

Categoría: Operadores sobre símbolos.

Descripción: Se hace necesario la gestión de los símbolos variables creados durante la ejecución, lo que implica la asignación de valores a las variables que serán definidas y utilizadas por el contenido fuente dado por el usuario. El valor que es asignado a una variable puede ser cualquier tipo de dato contemplado, excepto clase de objetos. El valor asignado puede ser determinado a partir de cualquier expresión que tenga un valor asociado después de su ejecución. En esta operación el valor es asignado a la variable mediante una copia del mismo. La operación de asignación debe representar una expresión que toma como valor tras su ejecución el valor asignado.

Número: 0056

Nombre: Asignación por referencia.

Categoría: Operadores sobre símbolos.

Descripción: Se debe facilitar un mecanismo para que dos símbolos variables distintos referencien al mismo valor. Para ello se ha de facilitar una operación de asignación por referencia tal que el valor asignado no sea copiado sino referenciado.

Número: 0057

Nombre: Acceso a símbolo de dato compuesto.

Categoría: Operadores sobre símbolos.

Descripción: Dado a que existen tipos de datos compuestos que deben mantener su propia tabla de símbolos, se precisa de un mecanismo para acceder a estos a partir del propio dato. Este mecanismo deberá devolver el símbolo al que se accede, pudiéndose aplicar cualquiera de las operaciones sobre símbolos.

Número: 0058

Nombre: Acceso a última posición.

Categoría: Operadores sobre símbolos.

Descripción: La mayoría de datos compuestos consisten en un listado de elementos. Se hace necesario un mecanismo para referenciar el final de este listado. A esta referencia se le podrá asignar algún dato, lo que lo añadirá al final del listado.

Número: 0059

Nombre: AND lógico.

Categoría: Operadores lógicos.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica AND. Para ello se deberá tomar el valor lógico de cada uno de los operandos. La evaluación de la operación lógica AND debe ser de cortocircuito, tomándose el valor del último elemento evaluado. Así, aunque esta expresión se corresponde con un operador lógico, el valor de la misma será el del último elemento evaluado.

Número: 0060

Nombre: OR lógico.

Categoría: Operadores lógicos.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica OR. Para ello se deberá tomar el valor lógico de cada uno de los operandos. La evaluación de la operación lógica OR debe ser de cortocircuito, tomándose el valor del último elemento evaluado. Así, aunque esta expresión se corresponde con un operador lógico, el valor de la misma será el del último elemento evaluado.

Número: 0061

Nombre: NOT lógico.

Categoría: Operadores lógicos.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica NOT. Para ello se deberá tomar el valor lógico de su único operando y negarlo. La expresión deberá tomar un valor de tipo booleano tras realizarse la operación.

Número: 0062

Nombre: Vacío.

Categoría: Operadores lógicos.

Descripción: Se necesita de un operador que determine si un dato se considera vacío. Este operador tendrá un único operando y funcionará igual que el operador lógico NOT. El valor que tomará la expresión será lógico.

Número: 0063

Nombre: Es nulo.

Categoría: Operadores lógicos.

Descripción: Se necesita de un operador que determine si dato o expresión contiene el valor nulo.

Número: 0064

Nombre: Menor que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "menor que". Para ello se deberá tomar el valor aritmético de cada operando. La expresión deberá tomar un valor de tipo booleano tras realizarse la operación.

Número: 0065

Nombre: Menor o igual que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "menor o igual que". Para ello se deberá tomar el valor aritmético de cada operando. La expresión deberá tomar un valor de tipo booleano tras realizarse la operación.

Número: 0066

Nombre: Mayor que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "mayor que". Para ello se deberá tomar el valor aritmético de cada operando. La expresión deberá tomar un valor de tipo booleano tras realizarse la operación.

Número: 0067

Nombre: Mayor o igual que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "mayor o igual que". Para ello se deberá tomar el valor aritmético de cada operando. La expresión deberá tomar un valor de tipo booleano tras realizarse la operación.

Número: 0068

Nombre: Igual que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "igual que". La operación de igualdad debe ser independiente de los tipos de datos de los operandos, aplicándose en función del tipo de dato más completo que compartan. Por ejemplo si se compara un dato cadena que no representa un número racional con uno aritmético, como el tipo de dato común a ambos es el booleano, ambos tomarán su valor lógico para la comparación. Si ambos datos son tipos compuestos, se ha de comprobar mediante la operación de igualdad todos los elementos simples que lo componen por pares y de forma posicional. Como valor de la expresión se toma el valor booleano de la operación.

Número: 0069

Nombre: Idéntico que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "idéntico que". Esta operación se refiere a una operación lógica de igualdad pero contemplando además que los datos tengan el mismo tipo. Como valor de la expresión se debe tomar el valor booleano resultado de aplicar la operación.

Número: 0070

Nombre: Distinto que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "distinto que". Esta operación debe ser independiente de los tipos de datos de los operandos, aplicándose en función del tipo de dato más completo que compartan. Por ejemplo si se compara un dato cadena que no representa un número racional con uno aritmético, como el valor más completo que ambos pueden tomar es el booleano, tomarán su valor lógico para la comparación. Si ambos datos son tipos compuestos, se ha de comprobar mediante la operación de igualdad todos los elementos simples que lo componen por pares y de forma posicional. Como valor de la expresión se toma el valor booleano de la operación.

Número: 0071

Nombre: No idéntico que.

Categoría: Operadores de comparación.

Descripción: Se debe contemplar la expresión correspondiente a la operación lógica "no idéntico que". Esta operación se corresponde con la operación inversa de la operación "idéntico que". Como valor de la expresión se debe tomar el valor booleano resultado de aplicar la operación.

Número: 0072

Nombre: Suma.

Categoría: Operadores aritméticos.

Descripción: Se debe contemplar la expresión correspondiente a la operación aritmética "suma". Para realizar esta operación se deberá tomar el valor aritmético de cada operando. Tras realizarse la operación, el valor de la expresión deberá ser el resultado aritmético de la misma.

Número: 0073

Nombre: Resta.

Categoría: Operadores aritméticos.

Descripción: Se debe contemplar la expresión correspondiente a la operación aritmética "resta". Para realizar esta operación se deberá tomar el valor aritmético de cada operando. Tras realizarse la operación, el valor de la expresión deberá ser el resultado aritmético de la misma.

Número: 0074

Nombre: Producto.

Categoría: Operadores aritméticos.

Descripción: Se debe contemplar la expresión correspondiente a la operación aritmética "producto". Para realizar esta operación se deberá tomar el valor aritmético de cada operando. Tras realizarse la operación, el valor de la expresión deberá ser el resultado aritmético de la misma.

Número: 0075

Nombre: División.

Categoría: Operadores aritméticos.

Descripción: Se debe contemplar la expresión correspondiente a la operación aritmética "división". Para realizar esta operación se deberá tomar el valor aritmético de cada operando. El segundo operando debe ser distinto de 0. Si el segundo operando tiene valor aritmético 0 se deberá mostrar un error que informe del caso. Tras realizarse la operación, el valor de la expresión deberá ser el resultado aritmético de la misma.

Número: 0076

Nombre: Potencia.

Categoría: Operadores aritméticos.

Descripción: Se debe contemplar la expresión correspondiente a la operación aritmética "potencia". Para realizar esta operación se deberá tomar el valor aritmético de cada operando. Tras realizarse la operación, el valor de la expresión deberá ser el resultado aritmético de la misma.

Número: 0077

Nombre: Módulo.

Categoría: Operadores aritméticos.

Descripción: Se debe contemplar la expresión correspondiente a la operación aritmética "módulo". Para realizar esta operación se deberá tomar el valor aritmético de cada operando. El segundo operando debe ser distinto de 0. Si el segundo operando tiene valor aritmético 0 se deberá mostrar un error que informe del caso. Tras realizarse la operación, el valor de la expresión deberá ser el resultado aritmético de la misma.

Número: 0078

Nombre: Tamaño.

Categoría: Operadores aritméticos.

Descripción: Se precisa de algún mecanismo que dado un dato calcule el tamaño de este. Este operador calculará el tamaño dependiendo del tipo de dato del operando. Tras ejecutase la expresión el valor que tome será de tipo aritmético.

Lógico: Si es verdadero el tamaño es uno, si es falso será cero.

Aritmético: Tomará el número de dígitos decimales.

Cadena: El tamaño será el número de caracteres de la cadena.

Array: Para el tipo de dato array u otros derivados se el tamaño será el número de elementos contenidos en el mismo.

Otro tipo de dato: Se deberá dar un error de tipos.

Número: 0079

Nombre: Incremento y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, el valor de esta se de poder incrementar y reasignar. Para ello se tomará el valor aritmético de la variable, se incrementará en una unidad y se reasignará a la misma. El valor de la expresión será el de la variable incrementada.

Número: 0080

Nombre: Asignación e incremento.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, el valor de esta se de poder incrementar y reasignar. Para ello se tomará el valor aritmético de la variable, se incrementará en una unidad y se reasignará a la misma. El valor de la expresión será el de la variable antes de ser incrementada.

Número: 0081

Nombre: Decremento y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, el valor de esta se de poder decrementar y reasignar. Para ello se tomará el valor aritmético de la variable, se decrementará en una unidad y se reasignará a la misma. El valor de la expresión será el de la variable decrementada.

Número: 0082

Nombre: Asignación y decremento.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, el valor de esta se de poder decrementar y reasignar. Para ello se tomará el valor aritmético de la variable, se decrementará en una unidad y se reasignará a la misma. El valor de la expresión será el de la variable antes de ser decrementada.

Número: 0083

Nombre: Suma y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, al valor de esta se ha de poder sumar otra expresión y reasignarle el resultado. Para ello se tomará el valor aritmético de la variable, se le sumará el valor aritmético de la expresión y se reasignará a la variable el resultado. El valor de la expresión será el resultado de la suma aritmética.

Número: 0084

Nombre: Resta y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, al valor de esta se ha de poder restar otra expresión y reasignarle el resultado. Para ello se tomará el valor aritmético de la variable, se le restará el valor aritmético de la expresión y se reasignará a la variable el resultado. El valor de la expresión será el resultado de la resta aritmética.

Número: 0085

Nombre: Producto y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, al valor de esta se ha de poder multiplicar otra expresión y reasignarle el resultado. Para ello se tomará el valor aritmético de la variable, se calculará el producto con el valor aritmético de la expresión y se reasignará a la variable el resultado. El valor de la expresión será el resultado del producto aritmético.

Número: 0086

Nombre: División y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, al valor de esta se ha de poder dividir por otra expresión y reasignarle el resultado. Para ello se tomará el valor aritmético de la variable, se realizará la división por el valor aritmético de la expresión y se reasignará a la variable el resultado. La expresión no puede tener un valor aritmético de cero. El valor de la expresión será el resultado de la división aritmética.

Número: 0087

Nombre: Potencia y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, al valor de esta se ha de poder elevar a otra expresión y reasignarle el resultado. Para ello se tomará el valor aritmético de la variable, se elevará al valor aritmético de la expresión y se reasignará a la variable el resultado. El valor de la expresión será el resultado de la potencia.

Número: 0088

Nombre: Módulo y asignación.

Categoría: Operadores aritméticos.

Descripción: Dado un identificador u expresión que referencia a una dato variable, al valor de esta se ha de poder dividir por otra expresión y reasignarle el resto originado. Para ello se tomará el valor aritmético de la variable, se realizará la división por el valor aritmético de la expresión y se reasignará a la variable el resto obtenido. La expresión no puede tener un valor aritmético de cero. El valor de la expresión será el resultado de la operación módulo.

Número: 0089

Nombre: Conversión a aritmético.

Categoría: Operadores de conversión de tipo.

Descripción: Se ha de facilitar un operador que permita convertir un dato a tipo aritmético. Esta conversión se deberá realizar en función al tipo de dato origen y de la forma descrita en el requisito en el que se hace referencia al mismo. El valor que tomará la operación deberá ser el dato tras la conversión de tipos.

Número: 0090

Nombre: Conversión a lógico.

Categoría: Operadores de conversión de tipo.

Descripción: Se ha de facilitar un operador que permita convertir un dato a tipo lógico. Esta conversión se deberá realizar en función al tipo de dato origen y de la forma descrita en el requisito en el que se hace referencia al mismo. El valor que tomará la operación deberá ser el dato tras la conversión de tipos.

Número: 0091

Nombre: Conversión a cadena de caracteres.

Categoría: Operadores de conversión de tipo.

Descripción: Se ha de facilitar un operador que permita convertir un dato a tipo cadena de caracteres. Esta se conversión se deberá realizar en función al tipo de dato origen y de la forma descrita en el requisito en el que se hace referencia al mismo. El valor que tomará la operación deberá ser el dato tras la conversión de tipos.

Número: 0092

Nombre: Operador ternario.

Categoría: Operadores condicionales.

Descripción: Se ha de contemplar el operador "ternario". Este constará de tres expresiones que se denominarán "condición", "caso verdadero" y "caso negativo". Primero se deberá evaluar la expresión "condición" como un dato lógico, si esta es positiva el valor de la operación será el de la expresión "caso verdadero", si es negativa se tomará el de "caso falso".

Se debe facilitar formas del operador ternario simplificadas en las que falten algunos de los operandos. Así se contemplará el ternario que carecerá de "caso verdadero" tomándose en su lugar el valor de la expresión "condición" (sin alterar el tipo de dato). También el ternario que carece de "caso falso" tomándose en su lugar el valor de la cadena vacía.

Número: 0093

Nombre: Operador no falso.

Categoría: Operadores condicionales.

Descripción: Se ha de contemplar el operador "no falso". Este consistirá en una lista de expresiones que serán evaluadas de forma secuencial y lógica, tomándose como valor el de la primera expresión cuya evaluación sea positiva, o el valor lógico falso si no existe ninguna.

Número: 0094

Nombre: Operador concatenación.

Categoría: Operadores sobre cadena de caracteres.

Descripción: La expresión que simboliza una operación de concatenación precisa de dos operandos que serán tratados como cadena de caracteres. El valor que tomará la expresión será la cadena resultante de concatenar ambas.

Número: 0095

Nombre: Operador explode.

Categoría: Operadores sobre cadena de caracteres.

Descripción: El operador explode deberá tomar dos cadenas de caracteres como operandos denominadas “texto” y “separador”. El valor de la operación será el array resultante de separar la cadena “texto” en diferentes cadenas en función la cadena “separador”.

Número: 0096

Nombre: Operador implode.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Representa la operación inversa a explode. El operador implode deberá tomar como operandos un array denominado “listado” y una cadena denominada “separador”. El valor de la expresión será una cadena resultado de concatenar cada uno de los elementos de “listado” separados por la cadena “separador”.

Número: 0097

Nombre: Operador formato.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Se hace necesario un mecanismo que permita generar cadenas formateadas. Este deberá consistir en una cadena de caracteres denominada "formato" y un listado de expresiones. La cadena formato contendrá una serie de directivas de formato. Estas directivas serán sustituidas por el valor correspondiente, según posición, de la lista de expresiones. Cuando se realiza cada sustitución el valor es formateado según la directiva.

Las directivas de formato tienen el siguiente forma:

$$\%[operador][precisin][formato]$$

Los posibles operadores serán los siguientes:

+: Fuerza la impresión del símbolo + cuando se formatean números positivos.

ˆ : Convierte el caracteres a mayúsculas cuando se formatean cadenas de texto.

#: Añade el carácter 0x cuando se formatean números hexadecimales y el carácter 0 cuando se formatean octales.

La precisión se refiere al número de decimales que se imprimirán en el caso de formatear números o el número de caracteres en el caso de formatear cadenas.

El carácter de formato indica que tipo de formato se le dará al valor:

i|d: Número entero.

u: Sin signo.

f: Coma flotante.

%: Carácter %.

e: Notación científica.

o: Octal.

x: Hexadecimal.

s|c: Cadena de texto.

Número: 0098

Nombre: Operador de búsqueda de subcadena.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Este es un operador básico en el tratamiento de cadenas. Opera sobre dos operandos que serán tratados como cadenas de caracteres, uno denominado "texto" y otro "subcadena". El operador toma como valor un dato aritmético relativo a la posición de la primera ocurrencia de "subcadena" dentro de "texto". Si no se encuentra ningún resultado el se tomará el valor -1. Adicionalmente se puede dar otro operando denominado "offset" que simbolice la posición dentro de "texto" a partir de la cual se comenzará a buscar.

Número: 0099

Nombre: Operador de remplazo de subcadena.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Se necesita de un mecanismo para buscar y remplazar subcadenas dentro de otra. El operador debe buscar las ocurrencias de una subcadena "búsqueda" en una cadena "texto", sustituyéndolas por una cadena de "remplazo". Este operador debe admitir el número máximo de sustituciones que se llevarán a cabo. Tras la ejecución debe tomar como valor la cadena resultante de sustituir en la cadena principal las ocurrencias de la subcadenas por la cadena de remplazo.

Adicionalmente la subcadena de "búsqueda" puede ser una expresión regular, en cuyo caso se buscará subcadenas que pertenezcan al conjunto de las palabras definido por la expresión regular.

Si se utiliza una expresión regular como patrón de búsqueda deberá ser posible utilizar en la cadena de remplazo parte de la subcadena que concuerda con la expresión regular. Para ello se forma la expresión regular mediante subexpresiones delimitadas por "()". En la cadena de remplazo se debe poder hacer referencia, de forma posicional, a las subcadenas correspondientes a cada una de las subexpresiones.

Número: 0100

Nombre: Operador de remplazo de subcadena mediante posiciones.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Se necesita de un mecanismo para buscar y remplazar subcadenas dentro de otra. El operador debe sustituir en una cadena "texto" la subcadena comprendida entre dos posiciones dadas por expresiones numéricas, sustituyéndo las subcadena correspondiente por una cadena de "remplazo". Tras la ejecución debe tomar como valor la cadena resultante de sustituir, en la cadena principal, la subcadena correspondiente a las posciones dadas, por la cadena de remplazo.

Número: 0101

Nombre: Operador conversión a mayúsculas.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Dada una cadena de caracteres se necesita de un operador que convierta todos los caracteres alfabéticos que la conforman en mayúsculas. El valor que se tomará será la cadena resultante de la operación.

Número: 0102

Nombre: Operador conversión a minúsculas.

Categoría: Operadores sobre cadena de caracteres.

Descripción: Dada una cadena de caracteres se necesita de un operador que convierta todos los caracteres alfabéticos que la conforman en minúsculas. El valor que se tomará será la cadena resultante de la operación.

Número: 0103

Nombre: Operador reducir array.

Categoría: Operadores sobre array.

Descripción: Se necesita de un operador que dado un array y una función reduzca el array a un solo valor. La función de reducción deberá recibir como parámetro dos valores correspondiente al valor acumulado y al nuevo valor. La función de reducción se ejecutará por cada elemento del array (execpto para el primero) tomando el valor acumulado y el nuevo valor, y devolviendo el próximo valor acumulado. Como valor este operador tomará el valor de la reducción.

Número: 0104

Nombre: Operador creador de expresión regular.

Categoría: Operadores sobre expresiones regulares.

Descripción: Dada una cadena de caracteres se necesita de un operador que convierta esta en una expresión regular. El valor del operador será la expresión regular.

Número: 0105

Nombre: Operador comprobación de expresión regular.

Categoría: Operadores sobre expresiones regulares.

Descripción: Se precisa un operador que, dada una expresión regular y una cadena de caracteres, compruebe si esta pertenece al lenguaje definido por la expresión regular. El operador tomará como valor un dato de tipo lógico resultado de la operación.

Para que el resultado sea positivo la cadena debe pertenecer al conjunto de palabras delimitadas por la expresión regular. Si tan solo existe correspondencia parcial el resultado será negativo.

Número: 0106

Nombre: Operador de búsqueda estructurada.

Categoría: Operadores sobre expresiones regulares.

Descripción: Se precisa de un mecanismo que lleve a cabo una búsqueda estructurada, es decir, obtener una estructura de datos array condicionada por una expresión regular denominada "patrón de búsqueda" y una cadena de caracteres "texto" sobre la que se comprueba.

En la ejecución de este operador se deberá buscar en "texto" subcadenas que pertenezcan al conjunto definido por la expresión regular, originando un array con cada una de las coincidencias, que será el valor que tome la operación.

Adicionalmente la expresión regular podría estar formada por subexpresiones delimitadas por "()". En dicho caso se buscará en "texto" subcadenas que pertenezcan al conjunto delimitado por la expresión regular. Por cada subcadena encontrada se creará un array con las correspondencias de cada subexpresión. Cada uno de los arrays resultantes se deberán guardar en otro que será el valor que tome la operación.

Si se utiliza una expresión regular formada por subexpresiones los arrays correspondientes a cada subcadena deberán tener índices numéricos. Sin embargo debe darse la posibilidad de especificar una lista ordenada de cadenas claves para crear un array asociativo.

Además se ha de contemplar la búsqueda estructurada sobre un array de cadenas "texto".

Número: 0107

Nombre: Operador fecha.

Categoría: Operadores de tiempo.

Descripción: Es necesario disponer de un operador que dé formato a la fecha/hora local. Este operador deberá tener como operando una expresión cadena de caracteres que contenga una serie de directivas de formato. El valor que tomará la operación será una cadena de caracteres que represente la fecha/hora en el formato dado. Las directivas de formato serán las siguientes:

%d: Día del més con dos dígitos.

%j: Día del mes sin ceros iniciales.

%l: Día de la semana de forma alfabética completa.

%D: Día de la semana de forma alfabética y con tres letras.

%w: Día de la semana de forma numérica (0-domigo,6-sábado).

%z: Día del año de forma numérica.

%F: Mes de forma alfabética.

%m: Mes de forma numérica con dos dígitos.

%n: Mes de forma numérica sin ceros iniciales.

%M: Mes de forma alfabética con tres letras.

%Y: Año con cuatro dígitos.

%y: Año con dos dígitos.

%a: Periodo del día (am/pm) en minúsculas.

%A: Periodo del día (am/pm) en mayúsculas.

%g: Hora en formato 12h sin ceros iniciales.

%G: Hora en formato 24h sin ceros iniciales.

%h: Hora en formato 12h con dos dígitos.

%H: Hora en formato 24h con dos dígitos.

%i: Minutos con dos dígitos.

%U: Segundos desde la Época Unix (1 de Enero del 1970 00:00:00 GMT).

%%: Carácter %.

Número: 0108

Nombre: Operador time.

Categoría: Operadores de tiempo.

Descripción: Se precisa de un operador que calcule el número de segundos desde la Época Unix (1 de Enero del 1970 00:00:00 GMT). Este operador no tendrá operandos y tomará el valor aritmético correspondiente.

Número: 0109

Nombre: Tipo puntero a fichero.

Categoría: Tipo de dato simple.

Descripción: El intérprete debe ser capaz de manipular ficheros, para ello se precisa de un tipo de dato que simbolice un puntero a un fichero del sistema de ficheros. Este tipo de dato no debe ser convertido a ningún otro tipo de dato ni viceversa. Además solo será tratado por algunos operadores dedicados.

No se tendrán en cuenta los ficheros binarios.

Número: 0110

Nombre: Abrir ficheros.

Categoría: Operadores sobre ficheros.

Descripción: Es necesario un operador que permita abrir ficheros para su manipulación. Este tendrá como operandos una cadena de caracteres que simbolice la ruta al fichero y otra que determine el modo en el que será abierto. El operador tomára como valor un dato de tipo puntero a fichero. Los posibles modos serán:

r: Lectura.

r+: Lectura y/o escritura.

w: Escritura truncando el contenido del fichero.

w+: Lectura y/o escritura truncando el contendio del fichero.

a: Escritura posicionando el puntero al final el fichero.

a+: Lectura y/o escritura posicionando el puntero al final del fichero.

Todos los modos a excepción de sólo lectura deberán crear el fichero si este no existe.

Número: 0111

Nombre: Cerrar ficheros.

Categoría: Operadores sobre ficheros.

Descripción: Es necesario un operador que permita cerrar ficheros abiertos a partir de un puntero al mismo. Se deberá finalizar cualquier flujo de datos abierto y el fichero quedará cerrado. Como valor se deberá tomar un dato de tipo lógico que determine si la operación se ha realizado correctamente.

Número: 0112

Nombre: Escribir en fichero.

Categoría: Operadores sobre ficheros.

Descripción: Se hace necesario un operador que, dado un dato de tipo puntero a fichero, pueda escribir datos en la posición referenciada por el mismo. Así este operador trabaja sobre dos operandos, un puntero a fichero y una cadena de caracteres que simbolizará el contenido a escribir. Como valor el operador toma el número de bytes que fueron escritos.

Número: 0113

Nombre: Leer de fichero.

Categoría: Operadores sobre ficheros.

Descripción: Se hace necesario un operador que, dado un dato de tipo puntero a fichero, lea desde la posición referencia por el mismo hasta un carácter de nueva línea, o bien un número de carácteres dado. Así el operador deberá tomar como valor una cadena de caracteres que represente el contenido leído.

Número: 0114

Nombre: Cambiar posición de puntero a fichero.

Categoría: Operadores sobre ficheros.

Descripción: Una operación básica sobre punteros a ficheros es desplazar este dentro del contenido del mismo. Para ello se precisa de un operador que, dado un puntero a fichero, cambie la posición de este dentro del propio fichero. Así la nueva posición deberá ser una expresión numérica que represente un offset relativo al principio del fichero, el final o la posicón actual del puntero. La expresión correspondiente al operador deberá tomar un valor booleano que determine si el cambio de posición se ha realizado correctamente.

Número: 0115

Nombre: Obtener la posición actual de puntero a fichero.

Categoría: Operadores sobre ficheros.

Descripción: Se necesita de un operador que dado un puntero a fichero tome el valor aritmético que represente la posición de este dentro del mismo.

Número: 0116

Nombre: Obtener contenido de un fichero.

Categoría: Operadores sobre ficheros.

Descripción: Se precisa de un operador que simplifique la tarea de obtener el contenido completo de un fichero, sin que sea necesario disponer de un puntero al mismo. Para ello se deberá facilitar una cadena de caracteres que simbolice la ruta completa del fichero. El operador tomará como valor una cadena de caracteres que contenga todo el contendio del fichero. En el caso de que el fichero no exista se deberá tomar como valor la cadena vacía.

Número: 0117

Nombre: Cadena como contenido de un fichero.

Categoría: Operadores sobre ficheros.

Descripción: Se precisa de un operador que simplifique la tarea de escribir una cadena de caracteres en un fichero, sin que sea necesario disponer de un puntero al mismo. Si el fichero existe su contenido deberá ser truncado, si no existe será creado. Este operador tendrá como operandos dos cadenas de caracteres que se correspondan con la ruta del fichero y la cadena a escribir. Como valor se tomára la cadena escrita.

Número: 0118

Nombre: Añadir cadena al contenido de un fichero.

Categoría: Operadores sobre ficheros.

Descripción: Se precisa de un operador que simplifique la tarea de añadir una cadena de caracteres al final de un fichero, sin que sea necesario disponer de un puntero al mismo. Si el fichero no existe será creado. Este operador tendrá como operandos dos cadenas de caracteres que se correspondan con la ruta del fichero y la cadena a escribir. Como valor se tomára la cadena escrita.

Número: 0119

Nombre: Defición de función

Categoría: Funciones.

Descripción: Se necesita de un mecanismo que permita definir y nominar bloques de sentencias. Estos bloques podrán recibir unos valores de entrada y producir una salida. Las sentencias en el bloque podrán operar sobre los parámetros de entrada, representados por unos símbolos variables que tomarán distintos valores en cada ejecución. Tras interpretarse el bloque de sentencias se podrá tomar un valor considerado de salida.

La definicón de una función representará en si misma un dato, por lo que podrán formar parte de operaciones y otras expresiones. Una función se define mediante un bloque de sentencias, una lista de identificadores que nominan a los parámetros de entrada y un identificador que le da nombre a la propia función, aunque este último no debe ser necesario (funciones anónimas).

Número: 0120

Nombre: Llamada a función

Categoría: Funciones.

Descripción: Dada una función, se debe disponer de un mecanismo que permita la ejecución del bloque de sentencias que la forma, mediante el uso de unos valores concretos como parámetros de entrada, y con la posibilidad de tomar el valor de salida.

Una llamada a función se deberá componer de un identificador relativo a su definición, y una lista de expresiones que determinarán los valores de los parámetros. La llamada deberá ser en si misma una expresión que tomará como valor la salida de la función tras la ejecución.

Los valores de los parámetros se corresponderán con los parámetros de la definición de la función de forma posicional.

Número: 0121

Nombre: Valor de retorno

Categoría: Funciones.

Descripción: Se necesita de un mecanismo en forma de sentencia que, dada una función, determine el valor del salida que se tomará en la llamada a la misma. La sentencia return se compondrá de una expresión correspondiente al valor salida. Al ser interpretada esta sentencia la ejecución de la función deberá finalizar y esta tomará el valor de la expresión dada.

Número: 0122

Nombre: Valores de parámetros por defecto.

Categoría: Funciones.

Descripción: Dada la definición de una función, debe existar un mecanismo para que los parámetros de esta puedan tener valores por defecto. Estos valores serán asignado a los parámetros cuando en una llamada a función no sean determiidados.

Como la correspondencia entre parámetros en una llamada a función se hace de forma posicional, los valores por defecto deberán ser especificados desde el final de la lista de parámetros hasta el inicio.

Número: 0123

Nombre: Parámetros por valor.

Categoría: Funciones.

Descripción: Cuando una función es ejecutada todos los símbolos variables que se definan y utilicen deben tratarse de forma local al bloque de sentencias de la función. De esta forma los símbolos variables definidos fuera de la función no serán accesibles desde el cuerpo de la misma y viceversa. Cuando se realice una llamada a función los valores de los parámetros deben ser copiados a los símbolos variables correspondientes.

Número: 0124

Nombre: Parámetros por referencia.

Categoría: Funciones.

Descripción: Se necesita de un mecanismo que permita que los parámetros de una función referencien valores definidos fuera del cuerpo de la misma. De esta forma se podrá acceder y/o modificar datos externos a la función.

Cuando en una llamada a función se especifiquen expresiones que sean símbolos variables como algunos de sus parámetros, si estos se definieron en la función como parámetros por referencia, el valor del símbolo en la llamada será referenciado por el símbolo correspondiente de la función.

Número: 0125

Nombre: Función lambda.

Categoría: Funciones.

Descripción: Se debe dar la posibilidad de crear funciones anónimas. Estas funciones carecerán de nombre y normalmente se utilizarán en la asignación de variables, como parámetros de otras funciones o como valor de retorno. Las funciones lambda deberán ser en si misma una expresión que toma como valor el dato correspondiente a la función.

Número: 0126

Nombre: Función lambda simple.

Categoría: Funciones.

Descripción: Se debe de facilitar un mecanismo para crear funciones lambdas simples, que solo consten de una lista de parámetros y de una única expresión que será devuelta y que constiturá el cuerpo de la función.

Número: 0127

Nombre: Referencia a función.

Categoría: Funciones.

Descripción: Se debe facilitar un mecanismo para referenciar funciones ya creadas, de forma que puedan ser asignadas a variables, pasadas como parámetros o devueltas como valor de retorno.

Número: 0128

Nombre: Currificación de funciones.

Categoría: Funciones.

Descripción: Se debe contemplar la currificación de funciones, para ello se debe poder definir funciones como valor de retorno de otras. Cuando la función devuelta sea ejecutada esta debe tener en cuenta el contexto en el que se definió.

Número: 0129

Nombre: Aplicacion parcial.

Categoría: Funciones.

Descripción: Se debe facilitar un mecanismo que permita, a partir de una función, obtener otra equivalente donde se ha dado valor a un subconjunto de los parámetros.

Número: 0130

Nombre: Decoradores.

Categoría: Funciones.

Descripción: Se debe facilitar un mecanismo para definir decoradores. Un decorador será un tipo especial de función. Al igual que una función se define mediante un identificador que lo nomina, una lista de parámetros y un bloque de sentencias.

A diferencia de las funciones ordinarias, la llamada a un decorador deberá tener como parámetro una función que será decorada, como resultado se deberá obtener una función que tendrá las siguientes características:

- La lista de parámetros que admite será la misma que la lista con la que se definió el decorador
- El bloque de sentencias será el del decorador pero haciendo uso de la función que ha sido decorada

Se debe facilitar un mecanismo para referenciar la función que se va a decorar dentro del decorador. Para ello se utilizará la función de contexto.

Número: 0131

Nombre: Función de contexto.

Categoría: Funciones.

Descripción: Se debe facilitar un mecanismo para acceder a la función de contexto. Esta será una función cuyo valor dependerá del contexto en el que se ejecute:

- En el primer nivel de ejecución la función de contexto no estará definida.
- En el cuerpo de una función será la propia función.
- En el cuerpo de un decorador será la función que se decorará.

Número: 0132

Nombre: Programación orientada a objetos.

Categoría: Clases de Objetos.

Descripción: Se debe contemplar una programación orientada a objetos basada en prototipos. Un objeto será creado directamente mediante sentencias o mediante copia de otros objetos.

Además también se deberá contemplar la creación de objetos mediante la instaciación de clases.

Las características de la programación orientada a objetos que se deberán contemplar son:

Abstracción: Un objeto por si mismo representará una entidad abstracta que podrá tener cierta funcionalidad asociada, disponer de atributos que establezcan su estado interno o comunicarse con otros objetos.

Encapsulamiento: Un objeto podrá contener todos los elementos correspondiente a su definición, estado y funcionalidad.

Principio de ocultación: Un objeto podrá tener atributos y/o métodos privados, de forma que sólo sean accesibles desde el propio objeto.

Polimorfismo: Se debe permitir a objetos de distinto tipo se le pueda enviar mensajes sintácticamente iguales, de forma que se pueda llamar un método de objeto sin tener que conocer su tipo.

Herencia: Se debe contemplar la herencia simple entre clases de forma que una clase se pueda definir mediante otra.

Número: 0133

Nombre: Construcción de objetos.

Categoría: Clases de Objetos.

Descripción: Un objeto se debe pueder construir de la misma forma que un array. Algunas de las claves se corresponderán con funciones que representarán los métodos del objeto, mientras que otras se corresponderan con otros tipos de datos que representarán los atributos.

Número: 0134

Nombre: Copia de objetos.

Categoría: Clases de Objetos.

Descripción: Los objetos deberán ser copiados mediante la operación de asignación. También debe ser posible la asignación por referencia de objetos.

Número: 0135

Nombre: Definición de clases.

Categoría: Clases de Objetos.

Descripción: Las clases definirán tipos de objetos que tedrán métodos y atributos comunes. Una clase se construye mediante un identificador que le da nombre y un bloque de sentencias que contendrá una serie de funciones (métodos) y símbolos variables (atributos).

Número: 0136

Nombre: Elementos privados.

Categoría: Clases de Objetos.

Descripción: Se debe failitar un mecanismo para definir atributos y métodos de una clase de objetos como privados. Estos elementos solo serán accesibles desde métodos del propio objeto. Se deberá contemplar el acceso a estos elementos sobre objetos del mismo tipo dentro de metódos de la clase.

Número: 0137

Nombre: Elementos estáticos.

Categoría: Clases de Objetos.

Descripción: Se debe failitar un mecanismo para definir atributos y métodos de una clase de objetos pertenecientes a la propia clase. Estos elementos no serán trasladados a los objetos instanciados.

Número: 0138

Nombre: Herencia de clases.

Categoría: Clases de Objetos.

Descripción: Se debe de disponer de un mecanismo que permita establecer una relación de herencia entre unas clases dadas. Así será posible la definición de nuevas clases partiendo de otras. La clase derivará de otra extendiendo su funcionalidad y definición.

En la definición de una clase se debe de disponer de un mecanismo que permita especificar la clase que se extenderá. La nueva clase tendrá todos los atributos y métodos de la extendida y añadirá los suyos propios, pudiendo sobreescribirse los ya existentes.

Número: 0139

Nombre: Instanciación de clases.

Categoría: Clases de Objetos.

Descripción: Dada una clase, se debe de disponer de un mecanismo que permita crear objetos a partir de la misma. Para construir un objeto a partir de la instaniación de una clase se deben llevar las funciones y variables definidas en el cuerpo de la clase a métodos y atributos del objeto.

Una clase pude definir un método constructor que deberá ser llamado sobre el objeto recien creado cuando la clase es instanciada.

La instanciación se deberá realizar mediante un operador que, a partir de un identificador correspondiente a la clase y una lista de expresiones correspondientes a los parámetros del método constructor, tome como valor el objeto recien creado.

Número: 0140

Nombre: Acceso al objeto en ejecución.

Categoría: Clases de Objetos.

Descripción: Se debe de disponer de un mecanismo que permita acceder a los atibutos y métodos de un objeto desde la ejecución de un método del mismo. Este mecanismo, correspondiente a una expresión, deberá tomar como valor el objeto en ejecución.

Número: 0141

Nombre: Acceso al objeto en ejecución como clase padre.

Categoría: Clases de Objetos.

Descripción: Se debe de disponer de un mecanismo que permita acceder a los atibutos y métodos de la clase padre de un objeto desde la ejecución de un método del mismo. Este mecanismo, correspondiente a una expresión, deberá tomar como valor el objeto en ejecución, pero tomando como métodos y atributos los de la clase padre de la cual deriva.

Número: 0142

Nombre: Enlace estático en tiempo de ejecución.

Categoría: Clases de Objetos.

Descripción: Se debe de disponer de un mecanismo que permita acceder a los atibutos y métodos estáticos de una clase hija desde un método estático de la clase padre.

Número: 0143

Nombre: Sentencia with.

Categoría: Clases de Objetos.

Descripción: Se deberá facilitar un mecanismo que permitar establer un objeto como contexto. Así toda llamada a función que se haga, y cuya definición no exista, se deberá hacer como una llamada a un método del objeto. Esta sentencia se deberá construir a partir del objeto y un bloque de sentencias sobre el que se aplicará el contexto.

Número: 0144

Nombre: Conversión de tipos.

Categoría: Clases de Objetos.

Descripción: Se debe facilitar un mecanisno para que una clase u objeto pueda definir métodos que determinen como llevarse a cabo la conversión de las instancias a otros tipos de datos.

Número: 0145

Nombre: Ejecutar comando.

Categoría: Procesos.

Descripción: Se debe facilitar un operador que ejecute un comando dado. Para ello se deberá usar el interprete de comandos definido por el entorno del sistema operativo. Este operador contemplará un único operando correspondiente al comando a ejecutar. Como valor se deberá tomar la cedena de caracteres correspondiente a la salida del comando.

Número: 0146

Nombre: Evaluar cadena.

Categoría: Procesos.

Descripción: Deberá existir un operador que utilice el interprete para procesar una cadena de caracteres escrita en el léxico y con la sintaxis del propio lenguaje. Este operador tomará como valor una cadena de caracteres relativa a la salida generada por la interpretación.

Número: 0147

Nombre: Bifurcación de proceso.

Categoría: Procesos.

Descripción: Se necesita de un mecanismo que bifurque el flujo de ejecución mediante la creación de un proceso hijo. El intérprete deberá crear un proceso clonado de si mismo, cuya ejución prosigirá en el mismo punto. El operador de bifurcación tomará valor aritmético cero en el proceso hijo, mientras que en el padre tomará el valor aritmético correspondiete al identificador del proceso hijo.

Número: 0148

Nombre: Espera entre procesos.

Categoría: Procesos.

Descripción: Se necesita de un mecanismo que permita hacer que la ejecución de un proceso padre espere a que todos o algunos de sus hijos finalicen su ejecución. Así este podrá operar o no sobre un dato aritmético que referenciará al identificador de proceso del hijo que se ha de esperar. El valor que tomará consistirá en el código correspondiente a la señal de salida producida por el último proceso finalizado.

Número: 0149

Nombre: Obtener identificador de proceso.

Categoría: Procesos.

Descripción: Todo proceso tiene un identificador único en el sistema sobre el que se ejecuta. Se deberá disponer de un operador que tome el valor del identificador de proceso correspondiente al interprete.

Número: 0150

Nombre: Obtener identificador de proceso padre.

Categoría: Procesos.

Descripción: Debe existir algún mecanismo que permita obtener el identificador del proceso padre cuando el interprete se ejecuta como proceso hijo de otro.

Número: 0151

Nombre: Señales entre procesos.

Categoría: Procesos.

Descripción: Debe existir algún mecanismo que permita mandar señales entre procesos. Estas señales se corresponderán con señales UNIX. Para mandar una señal a un proceso se deberá dar un identificador de proceso y un entero correspondiente a la señal a enviar.

Número: 0152

Nombre: Manejador de señales a procesos.

Categoría: Procesos.

Descripción: Debe existir algún mecanismo que permita especificar una función que será ejecutada cuando el proceso reciba una determinada señal.

Número: 0153

Nombre: Llamar a función como proceso

Categoría: Operadores sobre procesos.

Descripción: Se precisa de operador que mediante una función y un listado de parámetros realice una llamada a la misma mediante la creación de un proceso hijo.

Número: 0154

Nombre: Parámetros al programa.

Categoría: Entrada.

Descripción: Se debe facilitar un mecanismo para que el contenido fuente pueda recibir parámetros de entrada desde la invocación a su interpretación. Estos parámetros deberán ser copiados a símbolos variables accesibles desde el contenido fuente. Adicionalmente se tratará otro parametro que se corresponderá con el número de parametros dados.

Número: 0155

Nombre: Variables de entorno.

Categoría: Entrada.

Descripción: Se necesita de un mecanismo para acceder a las variables de entorno definidas en el sistema operativo.

Número: 0156

Nombre: Generador de array.

Categoría: Generadores de expresiones.

Descripción: Se necesita de un mecanismo que sea una expresión por si mismo y que permita generar arrays desde una sentencia iterativa. Este mecanismo se formará mediante una expresión seguida da una sentencia for. La expresión será ejecutada tras iteración del bucle y será asignada como último elemento de un array. Al final de la ejecución la expresión tomará el valor del array generado.

Número: 0157

Nombre: Extensiones.

Categoría: Extensiones.

Descripción: La funcionalidad y características del intérprete deben ser extensible mediante módulos dinámicos. Estos módulos añadirán sentencias, operadores y demás elementos propios de un lenguaje de programación.

Para que un extensión pueda ser utilizada se deberá cargar.

Número: 0158

Nombre: Carga de extensiones mediante fichero de configuración.

Categoría: Extensiones.

Descripción: Se debe facilitar un mecanismo que permita especificar un listado de extensiones que serán cargados al ejecutarse el interprete. Estos modulos serán especificados en un fichero de texto plano separados mediante saltos de línea. Toda ejecución del interprete conllevará la carga de las extensiones especificadas.

Número: 0159

Nombre: Carga de extensiones mediante sentencia.

Categoría: Extensiones.

Descripción: Se debe facilitar un mecanismo que permita cargar una extensión en tiempo de ejecución. Para ello se deberá facilitar la ruta del módulo correspondiente a la extensión como una cadena de caracteres. Tras la carga de la extensión las características de esta serán añadidas al interprete.

Número: 0160

Nombre: Extensión gettext.

Categoría: Extensión gettext.

Descripción: Se deberá facilitar a modo de extensión la funcionalidad y características de la biblioteca GNU de internacionalización (i18n), gettext.

Número: 0161

Nombre: Extensión mySQL.

Categoría: Extensión mySQL.

Descripción: Se deberá facilitar una extensión que amplie las capacidades del interprete mediante sentencias y recursos que permitan la interación con un sistema de gestión de datos mySQL.

3.4.2. Requisitos no funcionales

Descripción de otros requisitos (relacionados con la calidad del software) que el sistema deberá satisfacer: portabilidad, seguridad, estándares de obligado cumplimiento, accesibilidad, usabilidad, etc.

3.4.3. Reglas de negocio

En el desarrollo del sistema, hay que tener en cuenta las denominadas reglas de negocio, es decir, el conjunto de restricciones, normas o políticas de la organización que deben ser respetadas por el sistema, las cuales suelen ser cambiantes.

3.4.4. Requisitos de información

En esta sección se describen los requisitos de gestión de información (datos) que el sistema debe gestionar.

3.5. Alternativas de Solución

En esta sección, se debe ofrecer un estudio del arte de las diferentes alternativas tecnológicas que permitan satisfacer los requerimientos del sistema, para luego seleccionar (si procede) la herramienta o conjunto de herramientas que utilizaremos como base para el software a desarrollar.

3.6. Solución Propuesta

Si se ha optado por utilizar un software de base, debemos identificar y medir las diferencias entre lo que proporciona este software y los requisitos definidos para el proyecto.
El resultado de este análisis permitirá identificar cuáles de éstos requisitos ya están solventados total o parcialmente por el sistema base y cuales tendremos que diseñar e implementar la propuesta de solución.

Capítulo 4

Análisis del Sistema

Esta sección cubre el análisis del sistema de información a desarrollar, haciendo uso del lenguaje de modelado UML.

4.1. Modelo Conceptual

A partir de los requisitos de información, se desarrollará un diagrama conceptual de clases UML, identificando las clases, atributos, relaciones, restricciones adicionales y reglas de derivación necesarias.

4.2. Modelo de Casos de Uso

A partir de los requisitos funcionales descritos anteriormente, se emplearan los casos de uso como mecanismo para representar las interacciones entre los actores y el sistema bajo estudio. Para cada caso de uso deberá indicarse los actores implicados, las precondiciones y postcondiciones, los pasos que conforman el escenario principal y el conjunto de posibles escenarios alternativos.

4.2.1. Actores

En este apartado se describirán los diferentes roles que juegan los usuarios que interactúan con el sistema. Los actores pueden ser roles de personas físicas, sistemas externos o incluso el tiempo (eventos temporales).

4.3. Modelo de Comportamiento

A partir de los casos de uso anteriores, se crea el modelo de comportamiento. Para ello, se realizarán los diagramas de secuencia del sistema, donde se identificarán las operaciones o servicios del sistema. Luego, se detallará el contrato de las operaciones identificadas.

4.4. Modelo de Interfaz de Usuario

En esta sección se deberá incluir un prototipo de baja fidelidad o mockup de la interfaz de usuario del sistema. Además, es preciso elaborar un diagrama de navegación, reflejando la secuencia de pantallas a las que tienen acceso los diferentes roles de usuario y la conexión entre éstas.

Capítulo 5

Diseño del Sistema

En esta sección se recoge la arquitectura general del sistema de información, la parametrización del software base (opcional), el diseño físico de datos, el diseño detallado de componentes software y el diseño detallado de la interfaz de usuario.

5.1. Arquitectura del Sistema

En esta sección se define la arquitectura general del sistema de información, especificando la infraestructura tecnológica necesaria para dar soporte al software y la estructura de los componentes que lo forman.

5.1.1. Arquitectura Física

En este apartado, describimos los principales elementos hardware que forman la arquitectura física de nuestro sistema, recogiendo por un lado los componentes del entorno de producción y los componentes de cliente.

Se debe incluir un modelo de despliegue en el cual se describe cómo los elementos software son desplegados en los elementos hardware. También se incluyen las especificaciones y los requisitos del hardware (servidores, etc.), así como de los elementos software (sistemas operativos, servicios, aplicaciones, etc.) necesarios.

5.1.2. Arquitectura Lógica

La arquitectura de diseño especifica la forma en que los artefactos software interactúan entre sí para lograr el comportamiento deseado en el sistema. En esta sección se muestra la comunicación entre el software base seleccionado, los componentes reutilizados y los componentes desarrollados para cumplir los requisitos de la aplicación. También, se recogen los servicios de sistemas externos con los que interactúa nuestro sistema. Se debe incluir un diagrama de componentes que muestre en un alto nivel de abstracción los artefactos que conforman el sistema.

Existen diferentes patrones o estilos arquitectónicos. En los sistemas web de información es común la utilización del patrón Layers (Capas), con el cual estructuramos el sistema en un número apropiado de capas, de forma que todos los componentes de una misma capa trabajan en el mismo nivel de abstracción y los servicios proporcionados por la capa superior utilizan

internamente los servicios proporcionados por la capa inmediatamente inferior. Habitualmente se tienen las siguientes capas:

Capa de presentación (frontend) Este grupo de artefactos software conforman la capa de presentación del sistema, incluyendo tanto los componentes de la vista como los elementos de control de la misma.

Capa de negocio Este grupo de artefactos software conforman la capa de negocio del sistema, incluyendo los elementos del modelo de dominio y los servicios (operaciones del sistema).

Capa de persistencia Este grupo de artefactos software conforman la capa de integración del sistema, incluyendo las clases de abstracción para el acceso a datos (BD o sistema de ficheros) o a sistemas heredados.

Es común que a la capa de negocio y de datos de los sistemas web, se denomine conjuntamente como backend o modelo de la aplicación.
Opcionalmente, podemos disponer de un conjunto de artefactos software que pueden ser usados por elementos de cualquiera de las capas del sistema y que fundamentalmente proporcionan servicios relacionados con requisitos no funcionales (calidad).

5.2. Parametrización del software base

En esta sección, se detallan las modificaciones a realizar sobre el software base, que son requeridas para la correcta construcción del sistema. En esta sección incluiremos las actuaciones necesarias sobre la interfaz de administración del sistema, sobre el código fuente o sobre el modelo de datos.

5.3. Diseño Físico de Datos

En esta sección se define la estructura física de datos que utilizará el sistema, a partir del modelo de conceptual de clases, de manera que teniendo presente los requisitos establecidos para el sistema de información y las particularidades del entorno tecnológico, se consiga un acceso eficiente de los datos. La estructura física se compone de tablas, índices, procedimientos almacenados, secuencias y otros elementos dependientes del SGBD a utilizar.

5.4. Diseño detallado de Componentes

Para cada uno de los módulos funcionales del sistema debemos realizar un diagrama de secuencia, para definir la interacción existente entre las clases de objetos que permitan responder a eventos externos.

5.5. Diseño detallado de la Interfaz de Usuario

En esta sección se detallarán las interfaces entre el sistema y el usuario, incluyendo un prototipo de alta fidelidad con el diseño de la IU. Se definirá el comportamiento de las diferentes pantallas, indicando qué ocurre en los distintos componentes visuales de la interfaz cuando aparecen y qué acciones se disparan cuando el usuario trabaja con ellas.

Capítulo 6

Construcción del Sistema

Este capítulo trata sobre todos los aspectos relacionados con la implementación del sistema en código, haciendo uso de un determinado entorno tecnológico.

6.1. Entorno de Construcción

En esta sección se debe indicar el marco tecnológico utilizado para la construcción del sistema: entorno de desarrollo (IDE), lenguaje de programación, herramientas de ayuda a la construcción y despliegue, control de versiones, repositorio de componentes, integración contínua, etc.

6.2. Código Fuente

Organización del código fuente, describiendo la utilidad de los diferentes ficheros y su distribución en paquetes o directorios. Asimismo, se incluirá algún extracto significativo de código fuente que sea de interés para ilustrar algún algoritmo o funcionalidad específica del sistema.

6.3. Scripts de Base de datos

Organización del código fuente, describiendo la utilidad de los diferentes ficheros y su distribución en paquetes o directorios. Asimismo, se incluirá el script de algún disparador o un procedimiento almacenado, que sea de interés para ilustrar algún aspecto concreto de la gestión de la base de datos.

Capítulo 7

Pruebas del Sistema

En este capítulo se presenta el plan de pruebas del sistema de información, incluyendo los diferentes tipos de pruebas que se han llevado a cabo, ya sean manuales (mediante listas de comprobación) o automatizadas mediante algún software específico de pruebas.

7.1. Estrategia

En esta sección se debe incluir el alcance de las pruebas, hasta donde se pretende llegar con ellas, si se registrarán todas o sólo aquellas de un cierto tipo y cómo se interpretarán y evaluarán los resultados. También, se incluirá el procedimiento a seguir para las pruebas de regresión, esto es, la repetición de ciertas pruebas para comprobar que nuevos cambios que se vayan introduciendo no originen errores en el software ya probado.

7.2. Entorno de Pruebas

Incluir en este apartado los requisitos de los entornos hardware/software donde se ejecutarán las pruebas.

7.3. Roles

Describir en esa sección cuáles serán los perfiles y participantes necesarios para la ejecución de cada uno de los niveles de prueba.

7.4. Niveles de Pruebas

En este sección se documentan los diferentes tipos de pruebas que se han llevado a cabo, ya sean manuales o automatizadas mediante algún software específico de pruebas.

7.4.1. Pruebas Unitarias

Las pruebas unitarias tienen por objetivo localizar errores en cada nuevo artefacto software desarrollado, antes que se produzca la integración con el resto de artefactos del sistema.

7.4.2. Pruebas de Integración

Este tipo de pruebas tienen por objetivo localizar errores en módulos o subsistemas completos, analizando la interacción entre varios artefactos software.

7.4.3. Pruebas de Sistema

En esta actividad se realizan las pruebas de sistema de modo que se asegure que el sistema cumple con todos los requisitos establecidos: funcionales, de almacenamiento, reglas de negocio y no funcionales. Se suelen desarrollar en un entorno específico para pruebas.

Pruebas Funcionales

Con estas pruebas se analiza el buen funcionamiento de la implementación de los flujos normales y alternativos de los distintos casos de uso del sistema.

Pruebas No Funcionales

Estas pruebas pretenden comprobar el funcionamiento del sistema, con respecto a los requisitos no funcionales identificados: eficiencia, seguridad, etc.

7.4.4. Pruebas de Aceptación

El objetivo de estas pruebas es demostrar que el producto está listo para el paso a producción. Suelen ser las mismas pruebas que se realizaron anteriormente pero en el entorno de producción. En estas pruebas, es importante la participación del cliente final.

Parte III

Epílogo

En esta última parte quedarán recogidas las conclusiones y los manuales necesarios para el manejo de la aplicación resultado del desarrollo. Si se ha realizado algún tipo de evaluación de la solución proporcionada, más allá de las pruebas del sistema, también deberá venir recogida en un capítulo separado dentro de esta parte. Pueden consultarse diversos tipos de evaluaciones sobre sistemas de información en [**?**]: casos de estudio, análisis estático, análisis dinámico, simulación, experimento controlado, etc.

Capítulo 8

Manual de implantación y explotación

Las instrucciones de instalación y explotación del sistema se detallan a continuación.

8.1. Introducción

Resumen de los principales objetivos, ámbito y alcance del software desarrollado.

8.2. Requisitos previos

Requisitos hardware y software para la correcta instalación del sistema.

8.3. Inventario de componentes

Lista de los componentes hardware y software que se incluyen en la versión del producto.

8.4. Procedimientos de instalación

Procedimientos de instalación y configuración de cada componente hardware y software (base y desarrollado) para asegurar la correcta instalación y explotación del sistema, así como aquellos procedimientos necesarios de migración/carga de datos.

8.5. Pruebas de implantación

Descripción de las pruebas a realizar después de la instalación del sistema.

8.6. Procedimientos de operación y nivel de servicio

Procedimientos necesarios para asegurar el correcto funcionamiento, rendimiento, disponibilidad y seguridad del sistema: back-ups, chequeo de logs, etc. También, es preciso indicar claramente aquellas actuaciones precisas necesarias para el mantenimiento preventivo del sistema y así prevenir posibles fallos en el mismo.

Capítulo 9

Manual de usuario

Las instrucciones de uso del sistema se detallan a continuación.

9.1. Introducción

Resumen de los principales objetivos, ámbito y alcance del software desarrollado.

9.2. Características

Recopilación de las principales funcionalidades del sistema.

9.3. Requisitos previos

Requisitos hardware y software para el correcto uso del sistema.

9.4. Uso del sistema

Describir todos los aspectos necesarios para una utilización efectiva y eficiente del sistema por parte de los usuarios.

Capítulo 10

Conclusiones

En este último capítulo se detallan las lecciones aprendidas tras el desarrollo del presente proyecto y se identifican las posibles oportunidades de mejora sobre el software desarrollado.

10.1. Objetivos alcanzados

Este apartado debe resumir los objetivos generales y específicos alcanzados, relacionándolos con todo lo descrito en el capítulo de introducción.

10.2. Lecciones aprendidas

A continuación, se detallan las buenas prácticas adquiridas, tanto tecnológicas como procedimentales, así como cualquier otro aspecto de interés.
Resumir cuantitativamente el tiempo y esfuerzo dedicados al proyecto a lo largo de su desarrollo que escribir un sencillo 'he trabajado mucho en este proyecto'.

10.3. Trabajo futuro

En esta sección, se presentan las diversas áreas u oportunidades de mejora detectadas durante el desarrollo del proyecto y que podrán ser abarcadas en futuras versiones del software.

Los elementos aquí descritos deben estar en relación con lo relatado en el apartado de objetivos y alcance del proyecto descritos en la introducción.

Bibliografía

Información sobre Licencia

Incluir aquí la información relativa a la licencia seleccionada para la documentación y software del presente proyecto.