**Ejercicio 1: a) ​Tome una de las variables de la línea 3 del siguiente código e indique y defina cuales son sus atributos:**

**1. Procedure Práctica4();**

**2. var**

**3. a,i:integer**

**4. p:puntero**

**5. Begin**

**6. a:=0;**

**7. new(p);**

**8. p:= ^i**

**9. for i:=1 to 9 do**

**10. a:=a+i;**

**11. end;**

**12. ...**

**13. p:= ^a;**

**14. ...**

**15. dispose(p);**

**16. end​;**

* a,i: integer

nombre: a

alcance: 3-16

tipo: integer

L-value (tiempo de vida): automatica

r-value: de a es 0-1 y de i es 1-9

**b) ​Compare los ​atributo s de la variable del punto a) con los atributos de la variable de la línea 4. Que dato contiene esta variable?, que otra variable hay en este código?**

* p: puntero

nombre: p

alcance: 4-15

tipo: puntero

L-value: automática

r-value:

**Ejercicio 2: a. Indique cuales son las diferentes formas de inicializar una variable en el momento de la declaración de la misma.**

Alcance estático

* Llamado alcance léxico
* Define el alcance en términos de la estructura léxica del programa
* Puede ligarse estáticamente a una declaración explícita o implícita examinando el texto del programa, sin necesidad de ejecutarlo

Alcance dinámico

* Define el alcance del nombre de la variable en términos de la ejecución del programa
* Cada declaración de variable extiende su efecto sobre todas las instrucciones ejecutadas posteriormente, hasta que una nueva declaración para una variable con el mismo nombre es encontrado durante la ejecución.

**b. Analice en los lenguajes: Java, C, Phyton y Ruby las diferentes formas de inicialización de variables que poseen. Realice un cuadro comparativo de esta característica.**

* En Java las variables de crean y luego se declaran o se crean y se declaran en una sola sentencia

int x;

int x=8;

**Ejercicio 3:​ Explique los siguientes conceptos asociados al atributo l-valor de una:**

**a. Variable estática:** se ligan a celdas de memoria antes de que comience la ejecución del programa y permanece ligada a las mismas celdas hasta que acaba la ejecución del programa. C?

**b. Variable automática o semi estática:** el tiempo de vida de este tipo de variables termina cuando se termina el bloque que contiene la sentencia de alocación.

**c. Variable dinámica:** la ligadura de espacio se produce cuando se elabora la sentencia de declaración, exceptuando las que se ligan estáticamente.

**d. Variable semi dinámica:** son variables sin nombre cuyo almacenamiento se asigna y desasigna en tiempo de ejecución mediante la utilización por el programador de instrucciones específicas.

**De al menos un ejemplo de cada uno. Investigue sobre qué tipos de variables respecto de su l-valor hay en los lenguajes C y Ada.**

**Ejercicio 4:**

**a. ¿A qué se denomina variable local y a que se denomina variable global?**

* Variable local: son variables cuyas referencias se crean dentro del programa o subprograma.
* Variable global: son variables cuyas referencias se crean en el programa principal.

**b. ¿Una variable local puede ser estática respecto de su l-valor? En caso afirmativo dé un ejemplo**

Si, se puede. Podría ser por ejemplo una variable de tipo static dentro de una función:

void pruebaLocalEstatica() {

static int x;

}

**c. Una variable global ¿siempre es estática? Justifique la respuesta.**

Las variables globales casi siempre son estáticas, ya que su l-valor se establece en tiempo de compilación. El tiempo de vida de las mismas es desde el inicio del programa hasta que éste finaliza, y el alcance es en todo el programa, excepto en aquellos lugares que se redefine una variable localmente y se toma a estas como prioridad.

**d. Indique qué diferencia hay entre una variable estática respecto de su l-valor y una constante**

La diferencia es que una variable estática habla de su l-valor, que la celda de memoria es siemprel a misma hasta que finaliza el programa mientras que en una constante, la misma puede ser automatica por ejemplo.

**Ejercicio 6:**

No tendría el mismo comportamiento ya que la **x** en el contexto global, es accesible para func1 y para func2, si estuviese dentro de func1, func2 no podría acceder a la misma.

Tanto la variable global como una estatica dentro de una funcion, su alocacion duraria desde el inicio del programa hasta la finalizacon del mismo, por estar alocadas ambas estaticamente.

**Ejercicio 7:**

Todas **globales** a la clase Persona:

public long id

public string nombreApellido

public Domicilio domicilio

private string dni;

public string fechaNac;

public static int cantTotalPersonas;

**Local** al método getEdad():

public int edad=0;

***El identificador es local pero el valor es no local??***

public string fN =this.getFechaNac();

Todas **globales** a la clase Domicilio:

public long id;

public static int nro

public string calle

public Localidad loc;

**Ejercicio 8:**

**a)**

i: 4 a 15

h: 5 a 15

mipuntero: 3 a 12

**b)**

i: 4 a 15

h: 5 a 15

mipuntero: 3 a 15

**c)**

No presenta ningun error, ya que el r-valor de h es 3+3 (6), y no una referencia al puntero eliminado mipuntero.

**d)**

Hay 2 errores, el primero es que mipuntero ya fue desalocado por lo tanto no es accesible, y el segundo es que se deberia acceder haciendo una dereferencia en el mismo para acceder a su valor, agregando el ^, entonces seria mipuntero^.

**e)**

tpuntero tiene tiempo de vida de 2 a 5 y alcance de 2 a 6

**f)**

Todas las variables son estáticas, ya que su l-valor no puede cambiar.

**Ejercicio 9:**

**a)**

void tiempoVidaMayorAlcance() {

static int i; // El tiempo de vida sera hasta que se termine el programa

i = rand();

}

void main () {

tiempoVidaMayorAlcance();

// No puedo acceder a i desde aca

return 0;

}

**b)**

***Punteros siguen siendo accesibles despues de dispose/free, pensarE***

Esta bien, el tiempo de vida si no se hace new no existe. El alcance sigue existiendo como cualquier variable.

**c)**

void tiempoAlcanceIgual() {

int a = 50;// Se crea la variable a y dura hasta que se sale de la funcion

a = 25;

} x

La variable a vive desde el principio y es accesible a lo largo de toda la función hasta que la misma finaliza.

**Ejercicio 10:**

A menos que se encuentre una nueva definición en un bloque interno de “c”, que esto enmascaraba a la variable y se tomaría por prioridad a la que está dentro del bloque. Además nos podemos encontrar con sentencias “return”, lo que haría que se termine el procedimiento y que la variable no llegue al final del texto del procedimiento.

**Ejercicio 11:**

**a)**

**I)**

Falso, eso es conocido como identificador

**II)**

Falso ,eeo es el alcance de una variable

**III)**

Falso, eso es el l-valor de una variable

**IV)**

Verdadero.

**b)**

Conjunto de valores que puede tomar y conjunto de operaciones que se pueden realizar sobre esos valores.

**Ejercicio 12:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identificador | Tipo | r-valor | Alcance | Tiempo de Vida |
| main | automatico |  | 1-14 | 1-14 |
| a (línea 4) | automatica | basura | 4-14 | 1-14 |
| n(linea 4) | automatica | basura | 4-14 | 1-14 |
| p(linea 4) | automatica | basura | 4-7.3 / 8-14 | 1-14 |
| v1(linea 5) | automatica | basura | 5-14 | 1-14 |
| c1(linea 6) | automatica | 10 | 6-7.2 / 8-14 | 1-14 |
| procedure Uno | automatica | - | 7,5-14 | 7-7.6 |
| v2 (linea 7.2) | semidinamica | basura | 7.2-7.6 | 7-7.6 |
| c1 (linea 7.3) | automatica | basura | 7.3-7.6 | 7-7.6 |
| c2 (linea7.3) | automatica | basura | 7.3-7.6 | 7-7.6 |
| p (linea 7.4) | automatica | null | 7.4-7.6 | 7-7.6 |
| q (linea 7.4) | automatica | null | 7.4-7.6 | 7-7.6 |
| p^ | dinamica | basura | 7.4-7.6 | 7.5.3-7.5.6 |
| q^ | dinamica | basura | 7.4-7.6 | 7.5.4-7.5.8 |

***Static y global en 0 sino basura.***

**Ejercicio 13:**

Hay convenciones para los nombres de variables.

En algunos lenguajes se usan prefijos para indicar si son globales o no, para indicar su tipo y sus posibles r-valores.

Fuera de algunos lenguajes particulares, el nombre de una variable, o sea un identificador no afecta en ninguno de los 4 ítems mencionados: tiempo de vida, alcance, r-valor y tipo.