ALGORITMOS Y ESTRUCTURAS DE DATOS II

Ingeniería Técnica en Informática de Gestión

Ingeniería Técnica en Informática de Sistemas

CURSO 2002/03

Práctica 1

Introducción al Diseño Modular y creación de TAD

Objetivos

- Familiarizarnos con el uso de los TAD como base del diseño modular.
- Repasar la creación de unidades en Pascal.
- Valorar la importancia de documentar, de forma correcta, la especificación de un TAD.
- Valorar la importancia de proporcionar robustez a los TAD creados.

Enunciado

Se desea especificar e implementar el TAD *TVector* para manejar vectores de números enteros con las siguientes características:

- El nombre del tipo será **vectorEnt**.
- El tamaño de un vectorEnt es de 10 posiciones.
- En un vectorEnt se almacenará cualquier número entero. El valor cero (0) servirá para indicar que una posición está vacía.
- Se permiten elementos repetidos.
- Operación tamaño(v): devuelve el número de enteros de *v* distintos de 0.
- Operación inserta(v,i,n): se produce error si el valor del índice *i* no se encuentra entre los límites correctos (1..10)
- Operación asigna(v_1,v_2): si los dos operandos tienen distinto tamaño, se vacía v_1 y se crea de nuevo con el tamaño y las componentes de v_2. En caso contrario se asignan las componentes de v_2 en v_1 siempre que la componente de v_2 no sea la vacía. Por ejemplo, si $v_1[3]=6$ y $v_2[3]=0$, entonces $v_1[3]=6$ después de aplicar la operación.
- Operaciones suma(v_1,v_2,v_r) y multiplica(v_1,v_2,v_r): ambos operandos deben tener el mismo tamaño. En caso contrario no se realizará la operación, mostrando un mensaje de ERROR y devolviendo el vectorEnt v_r vacío. Además, si se suma o multiplica una posición cualquiera con una posición vacía, el resultado en esa posición en v_r será vacía.
- Operación capicúa(v_1,v_2): devuelve *verdad* si v_1 y v_2 son capicúa, y *falso* en caso contrario
- Tratamiento de errores: todas las operaciones mostrarán un mensaje de error cuando se produzcan excepciones. Los mensajes de error se mostrarán por pantalla con el formato: *ERROR: <mensaje_de_error>* y un salto de línea.

Las cabeceras de las operaciones, junto con una breve explicación, son las siguientes:

procedimiento crearVector(**var** V: vectorEnt);
crea un vectorEnt vacío de tamaño 10
función esVacio(V: vectorEnt): booleano;
devuelve verdad si el vectorEnt V está vacío y falso en caso contrario
función tamaño(V: vectorEnt): natural;
devuelve el número de posiciones que tiene ocupadas el vectorEnt V
procedimiento inserta(**var** V: vectorEnt; i: natural; n: entero);
inserta el entero n en la posición i del vectorEnt V
procedimiento suma(V1,V2: vectorEnt; **var** VR: vectorEnt);
suma, componente a componente, los vectores V1 y V2, dejando el resultado en VR
procedimiento multiplica(V1,V2: vectorEnt; **var** VR: vectorEnt);
multiplica, componente a componente los vectores V1 y V2, dejando el resultado en VR
procedimiento asigna(**var** V1: vectorEnt; V2: vectorEnt);
realiza la asignación de V2 en V1 según la explicación mostrada anteriormente
función capicua(V1,v2: vectorEnt): booleano;
devuelve verdad si V1 y V2 son capicúa y falso en caso contrario
procedimiento verVector(V: vectorEnt);
*muestra por pantalla el contenido del vectorEnt V, separando cada componente por un espacio en blanco (sin guiones, saltos de línea ni mensajes). Para las componentes vacías se mostrará la palabra **VACIO**, y si el vectorEnt está vacío se imprimirá el mensaje **TVECTOR VACIO**.*

Se pide

1. Especificar e implementar el TAD TVector en un módulo (unidad) independiente. El fichero donde se almacene el módulo se debe llamar **tvector.pas**

Evaluación de la práctica

La práctica se evaluará mediante un programa que hace uso del TAD TVector. Este programa utilizará la sintaxis definida para cada operación, por lo que **resulta de vital importancia que se respeten, inexcusablemente, las cabeceras de las operaciones, así como el nombre del fichero que almacena la unidad donde se ha diseñado el TAD**. Cada alumno podrá crearse su propio programa para comprobar y evaluar el funcionamiento del TAD diseñado (este programa usuario **NO** se entregará en la práctica). Un ejemplo de programa de comprobación puede ser el siguiente:

```
{ programa para comprobar el uso del TAD TVector }
program cTVector;
uses tvector,crt;
var V1,V2,VR: vectorEnt;

begin
  crearVector(V1); crearVector(V2);
  writeln('******  COMIENZO DEL PROGRAMA  ******'); writeln;

  write('V1= ');verVector(V1);writeln;
  writeln('Asignación de valores a V2');writeln;
  inserta(V2,1,-5); inserta(V2,11,5); inserta(V2,2,6);
  inserta(V2,3,0); inserta(V2,12,4); inserta(V2,7,7);
  write('V2= ');verVector(V2);writeln;

  writeln('Asignación de Vectores');writeln;
  asigna(V1,V2);
  write('V1= ');verVector(V1);writeln;

  writeln('Suma de V1 y V2');writeln;
  suma(V1,V2,VR);
  write('V1+V2= ');verVector(VR);writeln;

  writeln('******  FIN DEL PROGRAMA  ******'); writeln;
end.
```

cuyo resultado en pantalla sería:

```
******* COMIENZO DEL PROGRAMA *******
V1= TVECTOR VACIO

Asignación de valores a V2
ERROR: índice 11 no válido
ERROR: índice 12 no válido
V2= -5 6 VACIO  VACIO  VACIO  VACIO  7 VACIO  VACIO  VACIO

Asignación de Vectores
V1= -5 6 VACIO  VACIO  VACIO  VACIO  7 VACIO  VACIO  VACIO

Suma de vectores
V1+V2= -10 12 VACIO  VACIO  VACIO  VACIO  14 VACIO  VACIO VACIO
******* FIN DEL PROGRAMA *******
```

NOTAS SOBRE LA CALIFICACION DE LAS PRACTICAS

1. Toda práctica que no respete la sintaxis (nombre y perfil) de las operaciones, nombre de unidad u otro tipo de anomalía que no permita comprobar el TAD, de forma automática, mediante el programa usuario de chequeo del profesor, **se considerará SUSPENSA**.

2. Las prácticas se evaluarán una sola vez, por lo que es importante comprobar el correcto funcionamiento de la práctica antes de entregarla. Una vez evaluada una práctica y entregada al alumno, **NO** se podrá entregar de nuevo corregida.

3. **NO** se recogerá ninguna práctica más allá de la fecha límite establecida para cada una de ellas.

4. Las prácticas **SOLO** se recogerán, el día establecido, durante la clase de laboratorio.

Tiempo estimado de realización

2 clases (se entregará en la semana del 11 al 15 de noviembre en la clase de prácticas)

NORMAS PARA LA PRESENTACIÓN DE PRÁCTICAS

Cada práctica entregada deberá contener **OBLIGATORIAMENTE** la siguiente información:

- **Portada:** Contiene el título de la práctica, nombre y apellidos del alumno, grupo de prácticas al que pertenece (A1 .. A7) y titulación (I.T.I. de Gestión ó I.T.I. de Sistemas). Se debe realizar con la plantilla diseñada, que se puede descargar desde la página de la asignatura.

- **Indice:** Contiene los apartados de los que se compone la práctica, indicando el número de página en el que se encuentran.

- **Documento de definición de cada TAD:** Se hará especial hincapié en su correcta documentación. Para definir los tipos, se indicará su **nombre**, **descripción, características** y los **valores no admitidos** en el tipo. La documentación de las operaciones se realizará utilizando las cinco cláusulas explicadas en teoría, tal como se puede ver en los siguientes ejemplos:

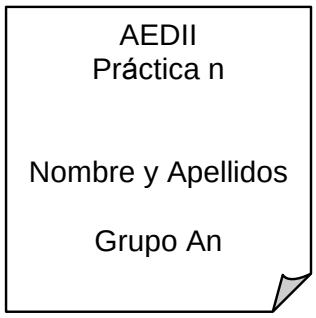
tipo: ABBEnt
descripción: Arbol Binario de Búsqueda de Números enteros. Estructura de tipo árbol binario de búsqueda en la que se almacenan números enteros.
características: Se permiten valores repetidos. La inserción de los valores repetidos se realizará por la izquierda.
valores no admitidos: Arboles con más de 50 nodos.

```
function Crear (nombre: cadena; edad, hijos: byte; var f: ficha): byte;
{
Parámetros:
    nombre: cadena de caracteres donde se almacena el nombre de la persona
    edad: número natural que indica la edad de la persona
    hijos: número natural que indica el número de hijos que tiene la persona
    f: variable de tipo ficha que se crea al ejecutarse con éxito la operación
Devuelve:
    0 ó 1. Si (edad < 16 y hijos > 0), devuelve 0. En caso contrario, la función devuelve
    1
Precondiciones:
                0 ≤ edad ≤ 110
                hijos ≤ 15
Efecto:
    Crea una ficha con los valores indicados en los argumentos
Excepciones:
    Si (edad < 16 y hijos > 0) no se crea la ficha f y se muestra un mensaje de error
}
```

Tipo de letra:

- Para la descripción del tipo y las cabeceras de las operaciones: Arial (12 pt.) con palabras reservadas en negrita
- Para la documentación de las operaciones: Arial (10 pt.) cursiva

- **Código fuente de los TAD (impreso y en disco):** Se tendrá en cuenta la correcta presentación, estructuración y eficiencia del código, no sólo su correcto funcionamiento. La etiqueta del disco debe contener la siguiente información:



- **Conclusiones y evaluación:** Breve opinión personal de la práctica realizada. Descripción de las situaciones "particulares" surgidas durante la realización de la misma, indicando las soluciones adoptadas en cada caso.

NOTA
La práctica se presentará en un portafolios de plástico, junto con un disco en su interior etiquetado con la información mostrada en la figura superior.