TIPOS ABSTRACTOS DE DATOS Y ESTRUCTURAS DE DATOS

Ejercicio 1:

Los números enteros, con la suma, la resta, la multiplicación y la división como operaciones básicas ¿es un tipo abstracto de datos?. Razona la respuesta.

Ejercicio 2:

Supongamos el tipo abstracto de datos 'Numero complejo', las operaciones que conocemos sobre él: suma, resta, multiplicación, división y conjugado de complejos, y dos operaciones adicionales de asignación y consulta:

Asigna (Real, Real) → Complejo

Que asigna a la parte real del número complejo 'x' y a la parte imaginaria 'y'

Valores (Complejo) → Real, Real

Que devuelve la parte real y la parte imaginaria del número complejo

Define al menos dos estructuras de datos válidas para su representación, e implementa las operaciones mencionadas sobre esas estructuras de datos en C++ (utiliza clases.)

Ejercicio 3:

Supongamos el tipo abstracto 'Matriz de números enteros'. ¿Qué operaciones básicas sería conveniente definir?

Implementa con clases de C++ el tipo abstracto 'Matriz de números enteros', y las operaciones que has definido para él.

Ejercicio 4

Supongamos el tipo abstracto de datos '**`Cadena_Caracteres`**', que tiene definida la siguiente operación:

`Concatenar (Cadena_Caracteres, Cadena_Caracteres)` ® **`Cadena_Caracteres`**

Escribe en Pseudocódigo y C++ una definición adecuada de la estructura de datos que podemos utilizar para representar este tipo de dato, e implementa la función utilizando esa representación.

Ejercicio 5:

Supongamos el tipo abstracto de datos '**`Fecha`**' que contiene día, mes y año. Las operaciones definidas sobre este tipo serán:

Crear (Entero, Entero, Entero) → Fecha

Que a partir de un cierto día, mes y año crea una fecha correcta. Si los valores de día, mes y/o año fueran incompatibles mostrará error.

Incrementar (Fecha, Entero) → Fecha

Que incrementa la Fecha en un cierto valor de días

Dia_de_la_semana (Fecha) → Entero

Que devuelve el entero correspondiente al dia de la semana (Lunes = 1, Martes = 2, ...)

a.- Definir al menos dos represntaciones válidas del TAD '**`Fecha`**' en C++.

b.- Implementa correctamente como una clase en C++ las operaciones definidas sobre el TAD '**`Fecha`**'.

Ejercicio 6:

Supongamos el TAD conjunto. Haz una especificación informal de este TAD, con las operaciones que creas conveniente.

Ejercicio 7:

Implementar en C++ el TAD conjunto capaz de almacenar 512 valores de tipo entero.

Ejercicio 8:

Diseñar el TAD número Racional y sus diferentes operaciones.

Escribe primer una especificación informal del tipo abstracto de datos, e implementa a continuación el TAD con clases de C++.

Ejercicio 9:

Supongamos el TAD *tabla* sobre el que tenemos definidas las operaciones:

```
ELIMINAR (Tabla, Posicion) -> Tabla
INSERTAR (Tabla, Posicion, Valor) -> Tabla
RECUPERAR (Tabla, Posición) -> Valor
TAMANO (Tabla) -> Entero
VACIO (Tabla) -> Boolean
```

a.- Realizar un subprograma en pseudocódigo '**`CribaEratostenes`**' que determine los números primos entre 1 y 100 mediante el método de la Criba de Eratostenes, y que utilice sobre el tabla sólo las operaciones definidas.

b.- Implementa una clase Tabla en C++ que tenga este comportamiento y estas operaciones, y escribe en C++ el subprograma '**`CribaEratostenes`**'.

Ejercicio 10:

Escribir un programa en pseudocódigo que calcule:

$$m = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad v = \frac{1}{n}\sum_{i=1}^{n} (x_i - m)^2 \qquad d = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$$

$$\text{siendo} \begin{cases} n = 20 \\ x_i = i \quad i = 1,2,\ldots,n \end{cases}$$

suponiendo la información guardada en tablas.

Ejercicio 11:

Indica breve y claramente cuáles son las diferencias existentes entre la idea de tipo de dato en un lenguaje de programación procedural y el concepto de tipo abstracto de datos.

Ejercicio 12:

Un tipo abstracto de datos especifica:

- La representación en memoria de la información
- La representación en memoria de la información y las operaciones asociadas
- Las operaciones válidas para un tipo de datos y sus propiedades
- La declaración de tipos en un lenguaje de programación

Ejercicio 13:

Escribir un programa que realice la suma de dos números enteros positivos muy grandes (máximo 200 cifras), utilizar para ello las tablas vistas en el ejercicio 9.

Ejercicio 14:

Supongamos el tipo abstracto de datos (T.A.D.) tabla de valores, con las siguientes operaciones definidas:

CREAR_TABLA ( ) → Tabla
ALMACENAR (Tabla , Indice , Valor) → Tabla
RECUPERAR (Tabla , Indice) → Valor
VALOR_DEFINIDO (Tabla, Indice) → Boolean

¿Cuál de las siguientes afirmaciones es cierta?

- Cualquier T.A.D. es independiente de su implementación, por lo que cualquier T.A.D. se puede representar mediante *arrays* y mediante punteros, excepto las tablas que son básicamente *arrays*.
- El T.A.D. tabla, sólo puede ser implementado con punteros.
- Cualquier T.A.D. es independiente de su implementación, por lo que cualquier T.A.D., incluso las tablas se puede representar mediante *arrays* y mediante punteros.
- Una tabla, definida con estas operaciones y estos axiomas no puede ser considerado un tipo abstracto de datos.

Ejercicio 15:

¿Por qué es interesante la utilización de tipos abstractos de datos en programación?

- Para poder utilizar clases en C++.
- Para separar claramente la implementación y representación de datos de su utilización.
- Para que los programas que escribamos sean más fáciles de compilar y ejecutar.
- Para reducir el número de instrucciones de los algoritmos y mejorar su eficiencia.

Ejercicio 16:

Supongamos un lenguaje de programación que tiene como tipo base, el tipo Pila, pero no tiene como tipo base ni el tipo vector ni punteros. ¿Seria posible la implementación del tipo abstracto de datos Cola, con todas sus operaciones en ese lenguaje de programación?

- Sí, pero la implementación del T.A.D. cola se complica.
- No, ya que sólo es posible implementar los T.A.D. mediante la utilización de vectores y punteros.
- Sí, pero habría que modificar las operaciones básicas de la cola
- No, ya que es imposible implementar colas mediante pilas.

Ejercicio 17:

Dada una especificación formal de un tipo abstracto de datos

- Según lo que especifique el T.A.D., habrá que hacer una implementación estática o dinámica.
- El T.A.D. no especifica nada acerca de la implementación.
- El T.A.D. especifica exactamentelas operaciones y la representación interna de los datos.
- El T.A.D. sólo especifica la representación. Las estructuras de datos especifican las operaciones y los axiomas.