Algoritmos y Estructuras de Datos II - 1° cuatrimestre 2015

Práctico 2

1. Dado el tipo

   type *person* = **tuple**
   name : **string**
   age : **nat**
   weight : **nat**

   escribí un algoritmo que calcule la edad y peso promedio de un arreglo a : **array**$[1..N]$ **of** *person*.

2. Escribí un algoritmo que dado dos punteros p, q : **pointer to int**, intercambie los valores referidos. Escribí otro algoritmo que intercambie los valores de los punteros. Si hubiera un tercer puntero r : **pointer to int** que inicialmente sea igual a p, ¿cuál sería el valor referido por r luego de los intercambios?

3. Un palíndromo es una palabra o frase que, salvo por los espacios, se lee igual de izquierda a derecha que de derecha a izquierda. Por ejemplo: "dábale arroz a la zorra el abad" es un palíndromo. Escribí un algoritmo que dada una secuencia s de letras determine, con la ayuda de una pila p de letras, si s es o no un palíndromo. Asumiendo que se conoce la cantidad de letras de s (sin contar los espacios), sus elementos deben recorrerse del primero al último una sola vez.

   Nota: Se puede asumir que las letras aparecen sin tildes.

4. El tipo de los valores de verdad es muy elemental pero no siempre es uno de los tipos concretos en los lenguajes de programación. Especificá el TAD *Booleano* incluyendo constructores (*Verdadero* y *Falso*), las operaciones usuales ($\wedge, \vee, \Rightarrow, \equiv, \neg$), y las ecuaciones que las describen.

5. Algunos lenguajes de programación no incluyen el tipo *lista* como tipo concreto. Especificá el TAD *Lista* de valores de tipo T a partir de los constructores usuales: [ ] para crear la lista vacía y $\triangleright$ para agregar un elemento al principio de una lista.

   Otras operaciones a incluir son: $\triangleleft$ para insertar un elemento por detrás, *tamaño* para observar la longitud de la lista, *cabeza* para observar el primer elemento de la lista, *cola* para eliminar el primer elemento de la lista, *es_vacia* para observar si la lista es vacia, *concat* para concatenar dos listas, y . y $\downarrow$ para observar el elemento que se encuentra en una posición dada, y la sublista en esa posición. Notar que algunas operaciones tienen restricciones.

6. Especificá el tipo abstracto *conjunto finito* (de elementos de tipo T). Debe tener (al menos) operaciones para obtener la unión, intersección y diferencia de dos conjuntos, determinar si un conjunto es vacío y si un elemento pertenece a un conjunto. Se puede utilizar como constructores *vacío* o $\emptyset$ (crea el conjunto vacío), *unitario* o $\{\_\}$ (crea un conjunto con un único elemento) y *union* o $\cup$ (une dos conjuntos).

7. El TAD *cola* es ampliamente usado para resolver una gran variedad de problemas. Este uso se puede extender aún mas si se incluyen operaciones para agregar, observar y eliminar elementos por ambos extremos. Una variante con estas características se denomina *cola doble* o *bicola*. Especificá este TAD que además de las 5 operaciones habituales del TAD *cola*, incluye operaciones para: devolver el último elemento, eliminar el último elemento, y agregar un elemento al comienzo de la bicola. Detallá los constructores, operaciones y ecuaciones necesarias.

8. Especificá el TAD *pila reversible*, que tiene los mismos constructores y operaciones que el TAD *pila*, y además la operación *invertir*.

   Ayuda: Puede ser conveniente introducir operaciones auxiliares para caracterizar la operación *invertir*.

9. Implementá el TAD *booleanos* del ejercicio 4 utilizando un número natural.

10. Implementá el TAD *pila reversible* (ejercicio 8) utilizando arreglos. ¿Se te ocurre una implementación de manera que todas las operaciones sean de orden constante?

11. Implementá el TAD *conjunto finito* del ejercicio 6 sobre un tipo ordenado (por ejemplo, conjunto de enteros) utilizando listas (abstractas) de manera que los elementos se mantengan ordenados y sin repetición.

12. Especificá una expendedora de café y chocolate como tipo abstracto. El TAD tendrá, además del constructor vacío, tres constructores más: uno para reponer (una dosis de) café, otro para reponer (una dosis de) chocolate y otro para reponer un vaso descartable. La máquina expende café (respectivamente chocolate) con sólo oprimir el botón correspondiente.

    Otras operaciones a incluir son: una que averigua si el dispositivo tiene café, otra que averigua si tiene chocolate, otra que averigua si tiene vasos. Finalmente, habrá dos operaciones de eliminación (una de una dosis de café y otra de una dosis de chocolate), que corresponden a la función de la máquina de expender café (o chocolate) al usuario. Ambas operaciones eliminarán también un vaso descartable. Estas operaciones sólo se aplican en el caso de haber café (respectivamente chocolate) y vasos.

    Se pide también especificar tres operaciones que devuelvan, respectivamente, el número de dosis de café, de dosis de chocolate y de vasos que dispone la máquina, y tres operaciones correspondientes a la reposición, que permitan agregar n dosis de café, n dosis de chocolate, y n vasos.

    Implementá el TAD utilizando tuplas de contadores (TAD visto en el teórico) y tuplas de naturales, explicando la diferencia de eficiencia entre ambas implementaciones.

13. Especificá un TAD *tablero* para mantener el tanteador en contiendas deportivas entre dos equipos (equipo A y equipo B). Deberá tener un constructor para el comienzo del partido (tanteador inicial), un constructor para registrar un nuevo tanto del equipo A y uno para registrar un nuevo tanto del equipo B. El tablero sólo registra el estado actual del tanteador, por lo tanto el orden en que se fueron anotando los tantos es irrelevante.

    Además se requiere operaciones para comprobar si el tanteador está en cero, si el equipo A ha anotado algún tanto, si el equipo B ha anotado algún tanto, una que devuelva verdadero si y sólo si el equipo A va ganando, otra que devuelva verdadero si y sólo si el equipo B va ganando, y una que devuelva verdadero si y sólo si se registra un empate.

    Finalmente habrá una operación que permita anotarle un número n de tantos a un equipo y otra que permita "castigarlo" restándole un número n de tantos. En este último caso, si se le restan más tantos de los acumulados equivaldrá a no haber anotado ninguno desde el comienzo del partido.

    Implementá el TAD de la manera que consideres más conveniente, justificando tu elección.

14. Implementá el TAD *bicola* del ejercicio 7 utilizando la idea de *arreglo circular*.

15. Para el TAD *lista* del ejercicio 5, realizá una implementación con punteros, utilizando la estructura de datos de *lista enlazada*. Realizá una segunda implementación, esta vez utilizando *arreglos*.

    Para cada una de las implementaciones, indicá cuáles de las siguientes operaciones tienen orden constante: devolver el i-ésimo elemento, devolver el primer elemento, devolver el último elemento, insertar un elemento en el k-ésimo lugar, insertar un elemento al principio y devolver la longitud de la lista.

16. Implementá el TAD cola utilizando como representación listas enlazadas circulares, es decir, una cola está dada por un puntero al último nodo y éste a su vez contiene un puntero al primer nodo, etc. Utilizá la siguiente estructura:

    type *node* = **tuple**
        value : *elem*
        next : **pointer to** *node*
    type *queue* = **pointer to** *node*

17. Un programa de computación simbólica permite a sus usuarios manipular *polinomios con coeficientes enteros*. Para ello se utilizará un TAD que entre sus operaciones incluye evaluar en la indeterminada x y devolver el coeficiente de grado k.

    (a) Especificá el TAD. Se puede definir como constructores uno que devuelve el polinomio nulo y un segundo que suma un monomio (de la forma ax^k) a un polinomio.

    (b) Realizá una implementación con arreglos, de manera tal que la posición k aloje el coeficiente de grado k.

    (c) Obtené una implementación en la cual el espacio requerido no dependa del grado. Por ejemplo, el polinomio $x^{9000} + 1$ debería ser representado mediante un estructura de escaso tamaño.

    (d) Compará las ventajas y desventajas de cada implementación.