Marco Antonio Velazquez Figueroa 17212193

Ejemplos de tipos de datos abstractos

1.- Ejemplo: arrays. Para insertar o eliminar elementos podría exigirse que se desplazaran los elementos de modo que no existieran huecos. Por otra parte, tiene la desventaja de tener que dimensionar la estructura global de antemano, problema propio de las estructuras estáticas. · Representación enlazada: da lugar a la lista enlazada ya estudiada, en la que el orden físico no es necesariamente equivalente al orden lógico, el cual viene determinado por un campo de enlace explícito en cada elemento de la lista. Ventaja: se evitan movimientos de datos al insertar y eliminar elementos de la lista. Podemos implementar una lista enlazada de elementos mediante variables dinámicas o estáticas (el campo enlace será un puntero o un índice de array): 1. Variables de tipo puntero: Esta representación permite dimensionar en tiempo de ejecución (cuando se puede conocer las necesidades de memoria). La implementación de las operaciones son similares a las ya vista sobre la lista enlazada de punteros. 2. Variables estáticas: se utilizan arrays de registros con dos campos: elemento de la lista y campo enlace (de tipo índice del array) que determina el orden lógico de los elementos, indicando la posición del siguiente elemento de la lista (simulan un puntero). Esta representación es útil cuando el lenguaje de programación no dispone de punteros o cuando tenemos una estimación buena del tamaño total del array.

2.- Ejemplo: Una lista formada por A,B,C,D,E (en este orden) puede representarse mediante el siguiente array de registros: Con esta implementación debemos escribir nuestros propios algoritmos de manejo de memoria (Asignar y free). Nuestra memoria para almacenar la lista de elementos será el array de registros. De él tomaremos memoria para almacenar un nuevo elemento, y en él liberaremos el espacio cuando un elemento se elimina de la lista. Por tanto, en realidad coexisten dos listas enlazadas dentro del array de registros, una de memoria ocupada y otra de memoria libre, ambas terminadas con el enlace nulo -1. De esta forma, tendremos dos variables: lista, que contiene el índice del array donde comienza la lista enlazada de elementos (lista = 0) y libre, que contendrá el índice del array donde comienza la lista de posiciones de memoria libres (libre = 1). Cuando queramos añadir un nuevo elemento a la lista, se tomará un hueco de la lista libre, disminuyendo ésta y aumentando la primera. Si queremos eliminar un elemento, realizaremos la operación contraria. Cuando no existe ningún elemento en la lista, sólo existirá la lista libre, que enlaza todas las posiciones del array, mientras que si la lista se llena, no tendremos lista libre.

Bibliografía: Diseño De Algoritmos. Universidad de Malaga.