Introducción:

Una de las aplicaciones más interesantes y potentes de la memoria dinámica y los punteros son las estructuras dinámicas de datos. Las estructuras básicas disponibles en C y C++ tienen una importante limitación: no pueden cambiar de tamaño durante la ejecución. Los arreglos están compuestos por un determinado número de elementos, número que se decide en la fase de diseño, antes de que el programa ejecutable sea creado.

En muchas ocasiones se necesitan estructuras que puedan cambiar de tamaño durante la ejecución del programa. Por supuesto, podemos hacer 'arrays' dinámicos, pero una vez creados, tu tamaño también será fijo, y para hacer que crezcan o diminuyan de tamaño, deberemos reconstruirlas desde el principio.

Las estructuras dinámicas nos permiten crear estructuras de datos que se adapten a las necesidades reales a las que suelen enfrentarse nuestros programas. Pero no sólo eso, como veremos, también nos permitirán crear estructuras de datos muy flexibles, ya sea en cuanto al orden, la estructura interna o las relaciones entre los elementos que las componen.

Las estructuras de datos están compuestas de otras pequeñas estructuras a las que llamaremos nodos o elementos, que agrupan los datos con los que trabajará nuestro programa y además uno o más punteros autoreferenciales, es decir, punteros a objetos del mismo tipo nodo.

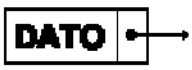
Una estructura básica de un nodo para crear listas de datos seria:

```
struct nodo {
   int dato;
   struct nodo *otronodo;
};
```

El campo "otronodo" puede apuntar a un objeto del tipo nodo. De este modo, cada nodo puede usarse como un ladrillo para construir listas de datos, y cada uno mantendrá ciertas relaciones con otros nodos.

Para acceder a un nodo de la estructura sólo necesitaremos un puntero a un nodo.

Durante el presente curso usaremos gráficos para mostrar la estructura de las estructuras de datos dinámicas. El nodo anterior se representará asi:



Las estructuras dinámicas son una implementación de TDAs o TADs (Tipos Abstractos de Datos). En estos tipos el interés se centra más en la estructura de los datos que en el tipo concreto de información que almacenan.

Dependiendo del número de punteros y de las relaciones entre nodos, podemos distinguir varios tipos de estructuras dinámicas. Enumeraremos ahora sólo de los tipos básicos:

- Listas abiertas: cada elemento sólo dispone de un puntero, que apuntará al siguiente elemento de la lista o valdrá NULL si es el último elemento.
- Pilas: son un tipo especial de lista, conocidas como listas LIFO (Last In, First Out: el último en entrar es el primero en salir). Los elementos se "amontonan" o apilan, de modo que sólo el elemento que está encima de la pila puede ser leído, y sólo pueden añadirse elementos encima de la pila.
- Colas: otro tipo de listas, conocidas como listas FIFO (First In, First Out: El primero en entrar es el primero en salir). Los elementos se almacenan en fila, pero sólo pueden añadirse por un extremo y leerse por el otro.
- Listas circulares: o listas cerradas, son parecidas a las listas abiertas, pero el último elemento apunta al primero. De hecho, en las listas circulares no puede hablarse de "primero" ni de "último". Cualquier nodo puede ser el nodo de entrada y salida.
- Listas doblemente enlazadas: cada elemento dispone de dos punteros, uno a punta al siguiente elemento y el otro al elemento anterior. Al contrario que las listas abiertas anteriores, estas listas pueden recorrerse en los dos sentidos.
- Arboles: cada elemento dispone de dos o más punteros, pero las referencias nunca son a elementos anteriores, de modo que la estructura se ramifica y crece igual que un árbol.
- Arboles binarios: son árboles donde cada nodo sólo puede apuntar a dos nodos.
- Arboles binarios de búsqueda (ABB): son árboles binarios ordenados. Desde cada nodo todos los nodos de una rama serán mayores, según la norma que se haya seguido para ordenar el árbol, y los de la otra rama serán menores.
- Arboles AVL: son también árboles de búsqueda, pero su estructura está más optimizada para reducir los tiempos de búsqueda.
- Arboles B: son estructuras más complejas, aunque también se trata de árboles de búsqueda, están mucho más optimizados que los anteriores.
- Tablas HASH: son estructuras auxiliares para ordenar listas.
- Grafos: es el siguiente nivel de complejidad, podemos considerar estas estructuras como árboles no jerarquizados.
- Diccionarios.

Al final del curso también veremos estructuras dinámicas en las que existen nodos de distintos tipos, en realidad no es obligatorio que las estructuras dinámicas estén compuestas por un único tipo de nodo, la flexibilidad y los tipos de estructuras sólo están limitados por tu imaginación como programador.