REPASO

YESSICA JANETH PABLO MARTINEZ

TIEMPO DE EJECUCIÓN

- **Tiempo de ejecución:** Este es el enfoque principal y se den como la longitud de tiempo en unidades requerida para la ejecución de un programa. Este puede depender de diversos factores, como son: la velocidad del hardware, características del sistema operativo y los dispositivos de entrada.

ESPACIO

- **Espacio.** Este enfoque es secundario y se representa por la cantidad de almacenamiento en computadora requerido para la ejecución de un programa. Cada computadora tiene una cantidad fija de espacio determinada por el tamaño de la memoria y la capacidad del disco.

ANÁLISIS DEL TIEMPO DE EJECUCIÓN

- La metodología de análisis de algoritmos la vamos a enfocar esencialmente a operaciones en estructuras de datos. Podemos plantear entonces el tiempo de ejecución de un algoritmo como una función sobre el tamaño de la entrada que establezca el número total de operaciones elementales que se llevan a cabo en el algoritmo

EJEMPLO 1

```
1 public static int ejem1(int x){
2     int y;
3     y = x + 1;
4     x = y * 2 ;
5     return y + x;
6 }
```

- Desglosemos el calculo del tiempo de ejecución para ejemplo1:
- la línea 2 hace una declaración de variable, lo cual constituye una operación elemental.
- la línea 3 hace un acceso a *x*, suma el valor con la literal 1 y el resultado lo asigna a *y*, con lo cual se tienen 3 operaciones elementales.
- la línea 4 hace un acceso a *y*, multiplica por 2 *y* el resultado lo asigna a *x*, con lo cual se tienen 3 operaciones elementales.
- la línea 5 hace un acceso a las variables *x* e *y*, suma sus valores y regresa el resultado, lo cual constituye 4 operaciones elementales.
- Por tanto, independientemente del valor de la entrada, siempre se ejecuta una secuencia de 1 + 3 + 3 + 4 = 11 operaciones elementales, lo cual es un numero constantes de operaciones elementales, decimos que el algoritmo toma tiempo constante.

EJEMPLO 2

```
1 public static int ejem2(int x){
2     if(x != 0 ) {
3         int y =  x + 2;
4         int z = x + y;
5         return z;
6     }else{
7         return 5;
8     }
9 }
```

- Para el ejemplo, el tiempo que se tarda en verificar la condición es constante, porque únicamente hace un acceso a x y una comparación con cero, son dos operaciones elementales. El máximo tiempo de las alternativas corresponde al de la primera, es decir, cuando se cumple la condición del ***if,*** pues la alternativa ***else*** requiere de una sola operación elemental, a saber, el retorno de la literal 5. Veamos entonces el tiempo de ejecución de la primera alternativa: en la segunda línea del método, se declara la variable y, se obtiene el valor de *x*, se suma con 2 y se asigna a *y* **(4 operaciones)** en la tercer línea, se declara la variable z, se obtiene el valor de *x*, se obtiene el valor de *y*, se suman *x* e *y* y luego se asigna el resultado de la suma a *z* **(5 operaciones).**
- Se accede al valor de ***z*** y se devuelve **(dos operaciones)**
- De aquí tenemos que el tiempo de ejecución de ejemplo2 es de 2 operaciones en la verificación del ***if***, aunado a 11 operaciones de la alternativa con mayor tiempo de ejecución, por lo que el tiempo de ejecución del método ejemplo2 es constante (13 operaciones).