```
/*
	1 Ejercicio 01
	Suma y Promedio de números.
*/
	int s = 0;				//1, asignacion
	double prom = 0;	//1, asignacion
	for (int i = 0; i < n; i++) {	//1 + n(1 + INTERNA + 2)
		s = s + A[i];				//3, acceso(1) + suma(1) +
asignacion(1)
		prom = s / (double)n;	//2, division(1) + asignacion(1)
	}
	printf("Suma:%d\nProm:%d", s, prom);	//1, impresion

/*
	Tiempo Detallado:
	1
	1
	1 + n(1 + 5 + 2)
	1
	= 4 + n(8)
	= 8n +4
	Tiempo Asintótico = O(n)
*/




/*
	2 Ejecicio 02
	Suma y Promedio de números (2.0).

*/
	int s = 0;				//1, asignacion
	double prom = 0;	//1, asignacion
	for (int i = 0; i < n; i++) {	//1 + n(1 + INTERNA + 2)
		s = s + A[i];				//3, acceso(1) + suma(1) +
asignacion(1)
	}
	prom = s / (double)n;	//2, division(1) + asignacion(1)
	printf("Suma:%d\nProm:%d", s, prom);	//1, impresion

/*
	Tiempo Detallado:
	1
	1
	1 + n(1 + 3 + 2)
	2
	1
	= 6 + n(6)
	= 6n + 6
	Tiempo Asintótico = O(n)
*/
```

```
/*
	3 Ejercicio 03
	Ordenamiento de N números del 1 al 100.

*/
	//int n = 10;		//For Test
	//int cont = 0;		//For Test
	for (int i = 0; i < n - 1; i++) {	//1 + (n-1)(1 + INTERNA_i + 2)
		for (int k = i + 1; k < n; k++) {	//2 + n/2(2 + INTERNA_k + 2) =>
2 + n/2(2 + 5 + 2) => 2 + n/2(9)
			//cout << "cont=" << ++cont << endl;	//For TEST
			if (vec[i] > vec[k]) {
				int aux = vec[i];	//2, acceso + asignacion
				vec[i] = vec[k];	//2, acceso + asignacion
				vec[k] = aux;			//1, asignacion
			}
		}
	}

/*
	Tiempo Detallado:
	1 + (n-1)(1 + 2 + n/2(9) + 2)

	= 1 + (n-1)(5 + 9n/2) = 1 + 5n-5 + 9n(n-1)/2
	= 5n - 4 + 9n^2 -9n/2
	= 9n^2 + n/2 -4
	Tiempo Asintótico = O(n^2)
*/




/*
	4 Ejercicio 04
	Ordenamiento de N números del 1 al 100 (2.0).
*/
	int frec[101] = { 0 };			//1, asignacion | frec[0] = 0 //primer
indice;
	for (int i = 0; i < n; i++) {	//1 + n(1 + INTERNA + 2) => 1 + n(1+4+2) => 1
+ 7n
		frec[vec[i]]++;	//4, acceso*2 + suma + asignacion
	}
	int pos = 0;				//1, asignacion
	for (int i = 0; i < 101; i++) {					//1 + 101(1
+ INTERNA + 2) => 1 + 101(1+ 8 +2) => 1 + 101(11) = 1112
		for (int k = 0; k < frec[i]; k++) {	//1 + 1 (2 + INTERNA + 2) => 1 +
(7) => 8
			vec[pos] = i;	//1, asignacion
			pos++;		//2, asignacion  + suma
		}
	}

	/*
		Tiempo Detallado:
		1
		1 + 7n
```

```
		1
		1112
		= 1115 + 7n
		Tiempo Asintótico = O(n)
	*/




/*
	5 Ejercicio 05
	Algoritmo raro - Infinito
*/
	int i = 0;                                                  //1, asignacion
	int sum = 0;                                          //1, asignacion
	while (i < 100) {                                     //100(INTERNA) =>
100(1+6n)
		if (i % 2 == 0) {   //MAX(if, else) => 1 + 6n
			for (int k = 0; k < n; k++) {    //1 + n(1 + 3 + 2) => 1 + 6n
				sum += vec[i];      //3, acceso + suma + asignacion
			}
		}
		else {
			for (int k = 0; k < i; k++) {    //1 + 100(1 + 3 + 2) => 601 =
constante
				sum += vec[i];      ///3, acceso + suma + asignacion
			}
		}
	}


	/*
		Tiempo Detallado:
		1
		1
		100(1+6n) = 100 +600n
		= 102 + 600n
		Tiempo Asintótico = O(n)
	*/




/*
	6 Ejercicio 06
	Factorial.

*/
	int fact = 1;                                         //1, asignacion
	for (int i = 2; i < n; i++) {          //1 + (n-2)(1 + INTERNA + 2) => 1 +
(n-2)(5) => 1 + 5n -10 => 5n - 9
		fact *= i;   //2, producto + asignacion
	}
	printf("Factorial: %d", fact);   //1, printf
```

```
	/*
		Tiempo Detallado:
		1
		5n - 9
		1
		= 5n - 7
		Tiempo Asintótico = O(n)
	*/
```

```
/*
	7 Ejercicio 07
	Buscar cadena de máximo 50 caracteres.

*/
	int pos = -1;                                                    //1,
asignacion
	for (int i = 0; i < n; i++) {                        //1 + n(1 + INTERNA + 2)
=> 1 + n(103) => 103n +1
		if (strcmp(vec[i], cadBuscar) == 0) {  //max 50
			pos = i;     //1
			break;
		}
	}


	/*
		Tiempo Detallado:
		= 53n + 1
		Tiempo Asintótico = O(n)
	*/
```