PRÓLOGO:
La filosofía del programador consiste, básicamente, en encontrar algoritmos genéricos a problemas genéricos, de tal forma que podamos trabajar con el máximo posible de casos dados, manteniendo la estructura del código sin alterar.

Es como dirían en FM (Para todo X….)

CAMBIO DE VARIABLE

Dado un double obtener su int: double x = 10.83; int y = int(x); // ahora x = 11;

Obtener el double de una división de ints: double x = 10 / double(3); // ahora x = 3.3~

Saber el valor ASCII de una letra: char c='s'; cout << int(c) << endl; // saca el codigo de 's';

ITINERANCIA VS RECURSIVIDAD:

Normalmente la recursividad nos permite trabajar un problema inverso (empezando por el final hasta el principio) o en el caso de los números, empezando por la cifra más alta.

Veamos pues un ejemplo de itinerancia y otro de recursividad:

ITINERANCIA:

```
int engreixa(int x) {
	int invertido;
	invertido = x % 10;
	x /= 10;
	int i = 1;
	while (x) {
		invertido = (invertido*10) + (x%10);
		x =  x / 10;
		++i;
	}
	int n;
	n = invertido % 10;
	invertido = invertido / 10;
	x = n;
```

```
        while (invertido or i > 1){
                int aux = invertido % 10;
                if(aux > n) n = aux;
                x = x*10 + n;
                invertido = invertido / 10;
                --i;
        }
        return x;
}
```

RECURSIVIDAD:

```
int engreixa(int x){
        if(x < 10) return x;
        int d = x%10;
        int aux = engreixa(x/10);
        if(aux%10 > d) d = aux%10;
        return ((aux* 10) + d);
}
```