Agenda diaria

- IDEs
- Punteros
- Stack
- ¿DRY?

- Visual Studio Community, Qt , Codeblocks
- Visual studio code, eclipse + CDT
- Online

La depuración es tu amiga

Punteros

referencia

```
int* pointVar;
int var = 5;

// assign address of var to pointVar
pointVar = &var;

// access value pointed by pointVar
cout << *pointVar << endl;   // Output: 5
```

Punteros a estructuras

```
#include <iostream>
using namespace std;

struct Distance {
    int feet;
    float inch;
};

int main() {
    Distance *ptr, d;

    ptr = &d;

    cout << "Enter feet: ";
    cin >> (*ptr).feet;
    cout << "Enter inch: ";
    cin >> (*ptr).inch;

    cout << "Displaying information." << endl;
    cout << "Distance = " << (*ptr).feet << " feet " << (*ptr).inch << " inches";

    return 0;
}
```

Operador ->

Since pointer ptr is pointing to variable d in this program, `(*ptr).inch` and `d.inch` are equivalent. Similarly, `(*ptr).feet` and `d.feet` are equivalent.

However, if we are using pointers, it is far more preferable to access struct members using the `->` operator, since the `.` operator has a higher precedence than the `*` operator.

Hence, we enclose `*ptr` in brackets when using `(*ptr).inch`. Because of this, it is easier to make mistakes if both operators are used together in a single code.

Precaución, ¡Punteros!

Derreferenciar (operador *) un puntero a *null* da problemas

Siempre que hay un *malloc*, debe haber un *free*

Siempre que hay un *new*, debe haber un *delete*

Siempre que hay un *new[]*, debe haber un *delete[]*

Tipos abstractos de datos

- Lineales:
- Pilas, colas, listas
- No lineales
- Árboles, Grafos

1er TAD

##Pila

Ejercicio

Dar la vuelta a una palabra utilizando un string (utiliza métodos)

Crea un programa que pueda evaluar una expresión aritmética expresada usando Notación Polaca Inversa [referencia](https://es.wikipedia.org/wiki/Notación*polaca*inversa)