Laboratorio de Computación II

Docentes

Angel Simón	Profesor	asimon@docentes.frgp.utn.edu.ar
Brian Lara	Profesor	blara@docentes.frgp.utn.edu.ar
Ariel Tapia	Jefe de trabajos prácticos	atapia@docentes.frgp.utn.edu.ar
Mariano Varela	Jefe de trabajos prácticos	mvarela@docentes.frgp.utn.edu.ar
Verónica Carbonari	Jefe de trabajos prácticos	vcarbonari@docentes.frgp.utn.edu.ar
Daniela Pinto	Ayudante de trabajos prácticos	daniela.pinto@alumnos.frgp.utn.edu.ar
Jeremías Roldán Zocco	Ayudante de trabajos prácticos	jeremias.roldan@alumnos.frgp.utn.edu.ar
Walter Alejandro Gonzalez	Ayudante de trabajos prácticos	walter.gonzalez@alumnos.frgp.utn.edu.ar

Contenidos

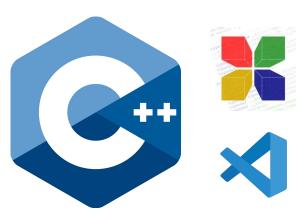
- Vectores y matrices
- Asignación dinámica de memoria
- Prog. Orientada a objetos
 - Encapsulamiento
 - Constructores y destructores
 - Herencia y composición
- Archivos

Evaluaciones

Examen Integrador

- Trabajo práctico de desarrollo grupal (hasta cuatro personas)
- Defensa grupal con preguntas y/o modificaciones individuales

Software





Alternativas:

Zinjai
Dev C++
Visual Studio
Visual Studio Code + plugin C++
Xcode



Breve repaso de Laboratorio I

Clasificación de variables

- Por tipo de dato
- Por dimensión
- Por alcance



Por tipo de dato

- Números enteros
- Números reales
- Texto
- Punteros



Por tipo de dato

- **Números enteros:** bool, short, int, long
- **Números reales:** float
- **Texto:** char, string
- Punteros



Por dimensión

- Variables simples
- Vectores
- Matrices
- Objetos



Clasificación de variables: Por dimensión

```
int edad;
                             char caracter;
Variables simples
                             float precio;
                             bool confirmar;
                             int cant[50];
                             char frase[100];
Vectores
                             float recaudacion2023[12];
                             bool vec[5];
                             int notas[50][6];
                             char nombres[100][41];
Matrices
                             float ingresos[12][5];
                             bool confirmar[3][3];
```



Por alcance

- Local
- Global



Clasificación de variables: Por alcance

Global

```
int numero;
int main(){
  numero=5;
  return 0;
void miFuncion(){
 numero = 10;
  return;
```

Local

```
void miFuncion(float var3){
   int var;
   char var2;
   return;
int main(){
   int var;
   return 0;
```



Punteros

```
int main(){
                                    Se declara una variable llamada p como
   int z;
                                    puntero a entero.
   z = 5;
                                    Se asigna la dirección de memoria de la
                                    variable z al puntero p. Se puede decir que p
                                    "apunta a" z.
   int *p;
      = &z;
                                    En la parte de memoria apuntada por p (o
                                    sea el contenido z) se asigna el valor 10.
      = 10;
   cout << *p;
                                    Se muestra por pantalla el contenido de
                                    donde está apuntando p. O sea, el valor de
   return 0;
                                    Z.
```



Funciones

```
int maximo(int a, int b){
  int c;
  if (a > b){
   c=a;
  else{
   c=b;
  return c;
```

Una función debe indicar un tipo de dato de devolución. **void** si no devuelve nada.

Debe tener un nombre.

Puede o no recibir parámetros.

La palabra clave **return** finaliza la función inmediatamente devolviendo el valor que se indique.



Pasaje de parámetros a funciones

Por valor

```
void mi_funcion ( int param_por_valor );
```

Por dirección

```
void mi_funcion ( int *param_por_direccion );
```

Por favor, no le digan referencia cuando en realidad es por dirección.

Por referencia

```
void mi_funcion ( int &param_por_referencia );
```



Ejercicio

A un grupo de personas candidatas a presidente se les asignó un número correlativo comenzando desde el 1. En total son siete candidatos.

Se desea un programa que compute los votos totales que obtuvieron los candidatos por municipio. Por cada municipio y candidato se registró:

- Código de municipio (1 a 280)
- Código de candidato (1 a 7)
- Cantidad de votos

No necesariamente existen votos para todos los municipios ni para todos los candidatos. La información no está agrupada ni ordenada. Se indica el fin de la carga de datos con un Código de municipio negativo. Calcular:

- El candidato que más votos ha obtenido. Indicar la cantidad de votos.
- La cantidad de municipios que hayan registrado más de 5000 votos en total.