Repaso de C Algo2Mendez

Rodolfo Albornoz
Máximo Palopoli
Julián Stiefkens

Agradecimientos a:
Gabriel Re
Tomas Ayala
Valentina Adelsflügel
Gabriel Bedoya (La abu)

Sintaxis

C

- Tipado estático
- Declaración de variables e inicialización
- Funciones y procedimientos
- Lenguaje Compilado

Python

- Tipado dinámico
- Lenguaje interpretado

Tipos de datos

- Char (1 byte)
- Int (4 bytes)
- Short (2 bytes)
- Long (8 bytes)
- Float (4 bytes)
- Double (8 bytes)
- Size_t (8 bytes)
- Bool (1 byte)
- Etc (unsigned…)

- Constantes
- Variables
- Vectores
- Structs--> Puede contener a todos los otros tipos de datos.

Declaración e inicialización de variables

C

- Int un_numero = 114;
- char una_letra = ‘C’;
- float otro_numero = 75,41;
- short numerito_cortito = 1;
- size_t ultimo_numero = 3;

Python

- numero1 = 1
- una_letra = ‘c’
- otro_numero = 75.41
- numerito_cortito = 1
- ultimo_numero = 3

Declaración e inicialización de constantes

Se declaran e inicializan al inicio del archivo

- const char PRIMER_LETRA = ‘A’;
- const size_t TAMANIO = 10;
- #define MAX_VECTOR 100
- #define NOMBRE “Messi”

```
const size_t TAMANIO = 10;

#define MAX_VECTOR 100
#define MAX_NOMBRE 50
#define NOMBRE "MESSI"
```

Define se utiliza para vectores y strings

Vectores

- int vector[MAX_VECTOR];
- int segundo_vector[2];

- vector [0] = 1;
- int primer_elemento_vector = vector[0];

Structs

```
typedef struct gato{
    char color_pelo;
    bool es_gordo;
    int anios;
    char nombre[MAX_NOMBRE];
    duenio_t esclavo;
    struct gato* hijos;
}gato_t;
```

```
36  typedef struct duenio{
37    int edad;
38    char nombre[MAX_NOMBRE];
39  }duenio_t
40
41  typedef struct gato{
42    char color_pelo;
43    bool es_gordo;
44    int anios;
45    char nombre[MAX_NOMBRE];
46    duenio_t esclavo;
47    struct gato* hijos;  // Puntero a struct gato
48  }gato_t;
```

Structs

```
// Manejo de struct

gato_t michi;    // De esta manera se acaba de crear un struct gato con basura en cada uno de sus campos

michi.anios = 2;
michi.es_gordo = false;
michi.color_pelo = 'N';
strcpy(michi.nombre, "Gato");     // Copia el string de la derecha y guarda la copia en el string de la izquierda

michi.esclavo.edad = 45;

gato_t neko = michi;

// Si se paso un struct por referencia

michi->anios = 2;
(*michi).anios = 2;
michi->esclavo.edad = 35;
```

If, else y else if

C

```c
if(var1 == 1){
    printf(“Algo1Mendez\n");
   }
else if(var1 == 2){
    printf(“Algo2Mendez\n");
   }
else{
    printf(“OtraCatedra\n");
   }
```

Python

```python
if var1 == 1:
    print(“Algo1Mendez")
elif var1 == 2:
    print(“Algo2Mendez")
else:
    print(“OtraCatedra")
```

While

C

```c
int n = 5
while(n > 0){
  n--;
  printf(“%d”,n);
}
```

Python

```python
n = 5
while n > 0:
  n -= 1
  print(n)
```

For

C

```c
int nums[MAX_VECTOR]={4,78,9,84};
for(int i = 0; i < MAX_VECTOR;i++){
    printf("%d\n",nums[i]);
}

int j = 90;
for(j; j > 0 ;j--){
    printf("%d\n",j);
}
```

Python

```python
nums = [4, 78, 9, 84]
for n in nums:
    print(n)
```

Switch

```
int nums[MAX_VECTOR] = {4,78};

switch(nums[i]){
 case 4:
   printf("4\n");
   break;
 case 78:
   printf("78\n");
   break;
 default:
    printf("No hay valores acá\n");
}
```

Cómo declarar una función (y procedimiento) y como llamarla

```
tipo_de_retorno nombre_función(parámetros)
{
    declaraciones/acciones
    return tipo_de_retorno
}

int suma(parámetros)
{
    hacer suma
}

void imprimir_numero(parámetros)
{
    imprimir
}
```