**LENGUAJES ELECTRÓNICOS**

Informe

**Autores:** Azorín Nicolás

**Curso:** 6° TEL

**Profesores:** Israel Pavelek y Antonella De Cía

**Institución:** Escuela Técnica Roberto Rocca.

**Informe N°:** 1

**Fecha de entrega:** 24/11/2022

25 de Noviembre – 2022

Contenido

[1. Introducción 3](#_Toc120261489)

[2. Listas Enlazadas (Lenguaje C) (desaprobado) 3](#_Toc120261490)

[4. Clases (En C++) (desaprobado) 6](#_Toc120261491)

[5. Diccionarios (En Python) (aprobado) 9](#_Toc120261492)

[6. Data Science 9](#_Toc120261493)

[7. Machine Learning 9](#_Toc120261494)

# Introducción

Este informe se trata sobre algunos de los temas dados en el año mostrando varios códigos de distintos lenguajes y explicados, los temas son que se van a explicar a continuación son:

* Listas Enlazadas
* Clases
* Diccionarios
* Data science
* Machine Learning

# Listas Enlazadas (Lenguaje C) (desaprobado)

Primero hay que comenzar preguntando, ¿Qué son las listas enlazadas? ¿Cómo funciona? las Listas Enlazadas son una estructura con tipos de datos lineales conectados entre sí a través de punteros. Contienen nodos los cuales se encargan de almacenar los datos ingresados, y también la dirección del siguiente nodo al que está conectado.



En esta imagen se puede ver mejor lo que se explicó anteriormente, la estructura con los datos conectados entre sí y los nodos con la dirección del siguiente al que está conectado.

(El código será explicado por partes para que sea todo mucho más claro).

**Parte 1 del código:**

Esta primer parte del código consiste en crear a una estructura llamada “Node” que dentro tiene una variable del tipo entero con el nombre: “data”. También dentro de la estructura tenemos un puntero que apunta a la estructura ya creada anteriormente, luego se crea otra variable del tipo entero que su función va a ser llevar la cuenta de las estructuras que se crearon. Una vez hecho eso realizo 3 prototipos:

1. Struct Node \* crear\_nodo (struct Node \*);

* El nombre de la función es “crear\_nodo”
* Recibe un valor del tipo struct Node \* (un puntero)
* Y devuelve un valor del tipo struct Node \* (también un puntero)

1. void ordenar (struct Node\*);

* Su nombre es “ordenar”
* no devuelve ningún valor
* Y recibe un valor del tipo struct Node\* (un puntero distinto al anterior);

1. void printNodes (struct Node\*);

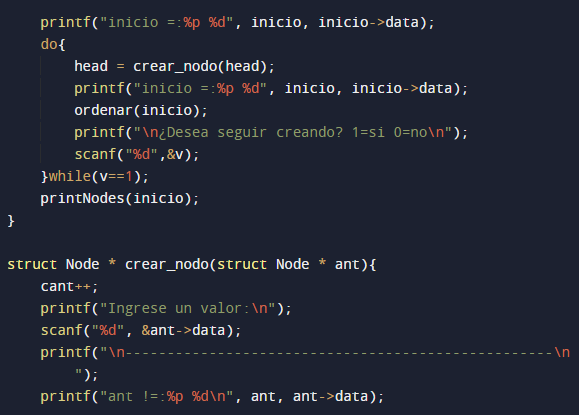
* El nombre de la función es “printNodes”
* no devuelve ningún valor
* Y recibe un valor del tipo struct Node\* (un puntero distinto a los anteriores);

Al terminar esto, pasamos al main del código el cual comienza con un puntero que puede ser cualquier tipo que este definido, a su vez se le asigna un “espacio” en la memoria del tamaño de la estructura, en este caso **struct Node.** Y luego lo que habíamos hecho anteriormente se iguala a otro puntero que apunta a **struct Node** que se llama “Nuevo”.

Al pasar esas 2 líneas de código creamos una nueva variable del tipo entero llamada “v” con un valor de 0 (aun no veremos su uso).

Y por último tenemos que *head* es igual a **crear\_nodo** el cual recibe una variable del tipo *TRUE* Node*.* (esto es porque al principio cuando declaramos *head* y la igualamos a inicio, nos damos cuenta que es un struct Node).

**Parte 2 del código:**

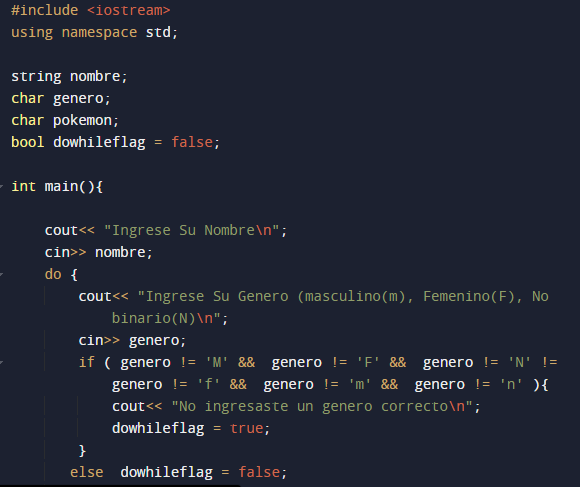


Ahora voy a explicar otra parte del código. En la primera línea solamente imprimimos lo que hay dentro del puntero *inicio,* y luego del **%d** imprimiría los valores de la data a donde apunta *inicio.*

Luego de esto tenemos el ciclo **Do While**, el cual sirve para crear el Nodo inicial, imprimir el valor del dato a donde apunta y luego pregunta, si queremos seguir creando mas nodos, al ingresar uno se cumplirá la condición del **while**, y así tener el ciclo y crear cuantos el usuario desee.

# Clases (En C++) (desaprobado)

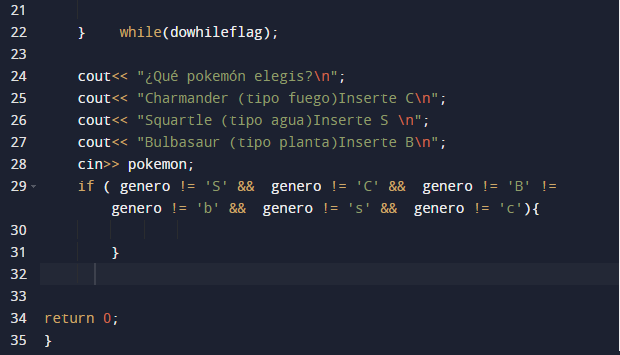
**Parte 1 del código:**



Para empezar el código se trata sobre los “Pokemones” tratando de “re-crear” el principio, cuando definís tú nombre, tú género y el Pokemón que vas a elegir:  (esto es lo que veríamos en el juego)

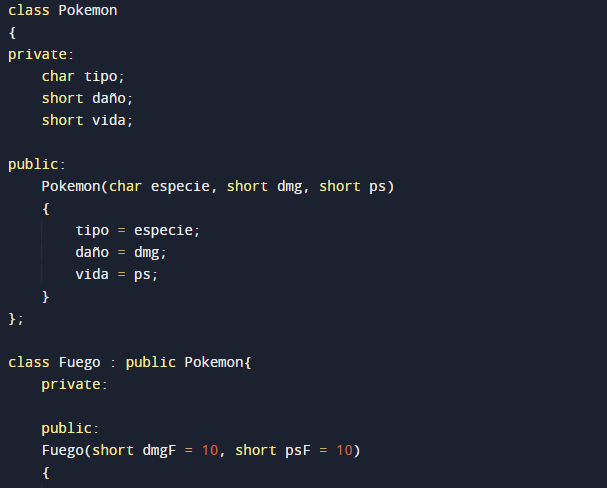
Pero en el código vemos una pequeña interfaz donde recibe nombre de la persona, genero de la misma y un checkeo del género (sí es correcto). Donde hace esto, es dentro del **Do While,** donde verifica la letra ingresada ya sea mayúscula o minúscula.

**Parte 2 del código:**



Acá vemos lo que sigue de la interfaz, donde nuevamente pregunta algo, pero esta vez sobre el Pokemón que tiene que elegir el usuario. De nuevo se checkea las letras si son mayúsculas o minúsculas, pero en este caso para elegir el Pokemón.

**Parte 3 del código:**



Primero comenzamos con la Clase de la cual heredan los siguientes:

-Fuego

-Agua

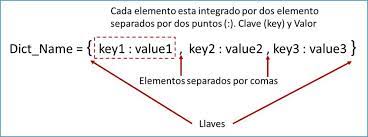
-Planta

Luego 3 tipos de variables, que van a servir para cada tipo de Pokémon tena un daño, porcentaje de vida y su tipo.

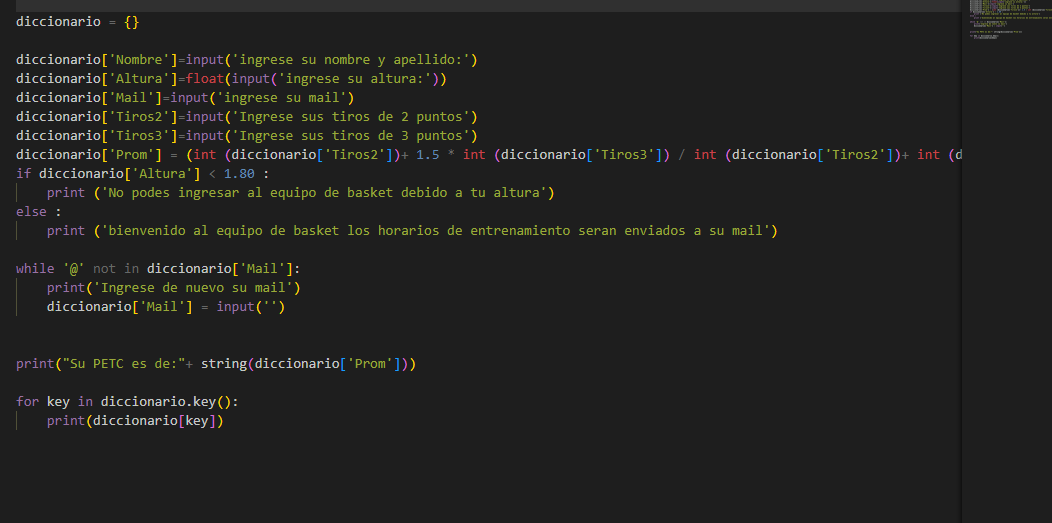
Más abajo tenemos escrita mas especifica la clase de “Fuego”, la cual tiene un constructor público, mostrando su daño y su porcentaje de vida.

# Diccionarios (En Python) (aprobado)

¿Qué son los diccionarios? Los diccionarios son una estructura de datos que permite almacenar cualquier tipo de información, desde cadenas de texto o caracteres hasta números enteros, con decimales, listas e incluso otros diccionarios. Al igual que sucede con un diccionario de lengua, los datos se encuentran ordenados utilizando una clave única para cada uno de ellos, lo que permite localizar cada uno de los datos de una forma muy rápida.



(Imagen representativa)



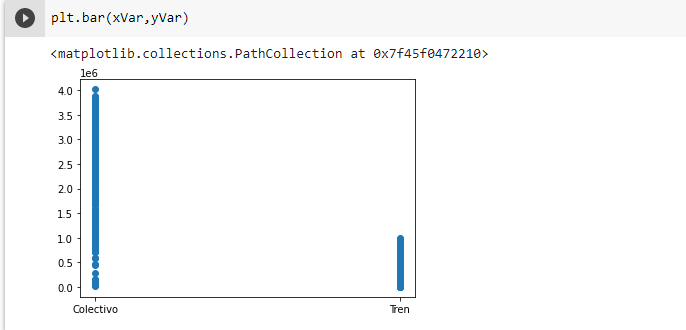
El código se trata sobre un una interfaz de un equipo de básquet la cual toma tus datos personales (nombre y apellido, su mail y su altura)

Resumidamente tenemos el diccionario (Explicado anteriormente), que su funcionamiento consiste en recibir nombre y apellido, la altura de la persona, la cantidad de tiros encestados de 2 puntos y 3 de 3 puntos, también luego verifica sí medís menos de 1.80 directamente te rechaza del equipo por la altura. Si cumplís la condición de medir más de 1.80 o 1.80 inclusive te va a mostrar tu promedio de tiro y te ingresará al equipo diciendo que te envió los horarios por mail.

# Análisis De Datos (con pandas y matplotlib)

Este ejercicio se trata sobre el análisis de datos de colectivos y trenes, su cantidad y los días en donde funciona, pero solamente tomamos la cantidad de cada uno y los comparamos en un gráfico de barra.

(gráfico donde se compara).



# Machine Learning

# Los principales algoritmos desarrollados en clase son: la regresión lineal, que intenta descubrir una función lineal para predecir datos, y la clasificación, que genera grupos de datos o clasificaciones a partir de la división binaria recursiva. Ambas estrategias predictivas se basan en el aprendizaje supervisado de la IA, que puede adaptarse por debajo (no entrenar lo suficiente), sobre adaptar (entrenar demasiado) o ajustarse de manera óptima. Esto sucede porque si le damos pocos datos de entrenamiento, el algoritmo de clasificación o función lineal resultante es demasiado simple para el sistema; por otro lado, si le damos demasiado, no podrá generalizar lo que tiene que predecir. datos. Si en lugar de brindarle demasiados o muy pocos datos, le brindamos la cantidad justa de datos de entrenamiento y prueba, podrá generar las mejores predicciones.

# ¿Qué aprendí?

# En “C”

|  |  |
| --- | --- |
| ¿Que aprendí? | Que me hubiera gustado aprender |
| Listas enlazadas | Algoritmos |
| Punteros a funciones |  |
| Estructuras |  |

# En “C++”

|  |  |
| --- | --- |
| ¿Que aprendí? | Que me hubiera gustado aprender |
| Clases | Métodos |
| Herencia |  |
|  |  |

# En “Python”

|  |  |
| --- | --- |
| ¿Que aprendí? | Que me hubiera gustado aprender |
| Diccionarios | Desarrollar algún videojuego |
| Graficar datos |  |
| Datasets |  |

# Machine Learning

|  |  |
| --- | --- |
| ¿Que aprendí? | Que me hubiera gustado aprender |
| Como funciona un IA | Como hacer un IA |
| Como aprende un IA |  |
|  |  |