https://github.com/nicobrch/ayudantias-udp

Programación Avanzada

Ayudantía 2: Programación Orientada a Objetos

https://github.com/nicobrch/ayudantias-udp

Constructor

Constructor

El constructor es un método base que puede estar o no estar incluido en nuestras clases.

Es el método que nos permite construir nuestra clase.

Constructor en código

Un constructor DEBE estar en el espacio público, de la siguiente forma:

```
class Usuario {
  private:
    string nombre;
  public:
    Usuario () {
    }
}
```

Constructor en código

Un constructor DEBE estar en el espacio público, de la siguiente forma:

```
class Usuario {
  private:
    string nombre;
  public:
    Usuario () {
    }
}
```

Constructor Vacio

Ya que no recibe ni asigna atributos.

Constructor no vacío

Un constructor DEBE estar en el espacio público, de la siguiente forma:

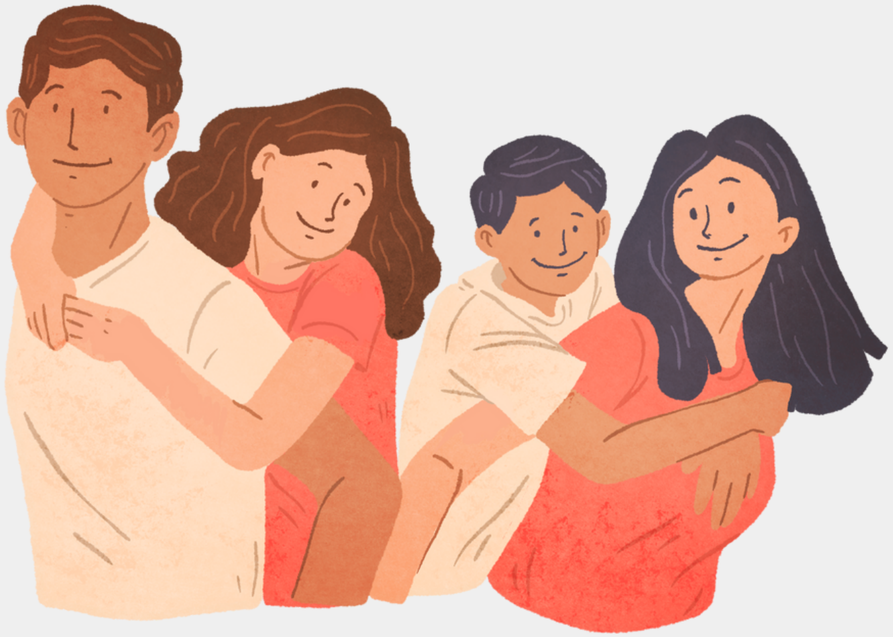
```
class Usuario {
  private:
    string nombre;
  public:
    Usuario (string n) {
      nombre = n;
    }
}
```

Constructor No Vacío

Recibe un parámetro y lo asigna a un atributo.

https://github.com/nicobrch/ayudantias-udp

Arreglos de Objetos



Arreglos de Objetos

Al igual que con los otros tipos de datos, podemos crear arreglos que tengan objetos, en vez de ints o strings.

Estos arreglos, igual que en programación I, tienen un tamaño definido.

¿Cómo funciona en código?

Supongamos que tenemos la clase Persona:

```
class Persona {
  private:
    string nombre;
    int edad;
  public:
    Persona (string n, int e) {
      nombre = n;
      edad = e;
    }
}
```

¿Cómo funciona en código?

Y ahora supongamos que tenemos la clase Familia:

```
class Familia {
  private:
    Persona *miembro[5];
  public:
    void agregarMiembro ( Persona *p ) {
    // codigo
    }
}
```

Clases dentro de Clases

Podemos crear una clase que tenga como atributos otras clases!
En el caso anterior, la clase familia tiene espacio para 5 "miembros", que son objetos de clase Persona.

Pero... ¿Cómo agregamos objetos al arreglo?

Agregar objetos al Arreglo

```
void agregarMiembro ( Persona *p ) {
  bool agregado = false;
  for ( int i = 0 ; i<5 ; i++ ){
    if (miembro[i] == NULL){
      miembro[i] = p;
      agregado = true;
    }
  }
  if (agregado == false){
    cout << "No hay cupos" << endl;
}
```

Agregar objetos al Arreglo

```
void agregarMiembro ( Persona *p ) {
 bool agregado = false;
  for ( int i = 0 ; i<5 ; i++ ){
    if (miembro[i] == NULL){
      miembro[i] = p;
      agregado = true;
    }
  }
  if (agregado == false){
    cout << "No hay cupos" << endl;
}
```

Variable Bool

Para comprobar si se agregó o no.

Agregar objetos al Arreglo

```
void agregarMiembro ( Persona *p ) {
  bool agregado = false;
  for ( int i = 0 ; i<5 ; i++ ){
    if (miembro[i] == NULL){
      miembro[i] = p;
      agregado = true;
    }
  }
  if (agregado == false){
    cout << "No hay cupos" << endl;
}
```

Ciclo FOR
hasta 5

Para recorrer el
arreglo de tamaño 5

Agregar objetos al Arreglo

```
void agregarMiembro ( Persona *p ) {
  bool agregado = false;
  for ( int i = 0 ; i<5 ; i++ ){
    if (miembro[i] == NULL){
      miembro[i] = p;
      agregado = true;
    }
  }
  if (agregado == false){
    cout << "No hay cupos" << endl;
}
```

Condicional IF NULL

Pregunto, ¿La posición i del arreglo está vacía?

Agregar objetos al Arreglo

```
void agregarMiembro ( Persona *p ) {
  bool agregado = false;
  for ( int i = 0 ; i<5 ; i++ ){
    if (miembro[i] == NULL){
      miembro[i] = p;
      agregado = true;
    }
  }
  if (agregado == false){
    cout << "No hay cupos" << endl;
}
```

Se cumple el IF

Agrego miembro y marco como agregado

Agregar objetos al Arreglo

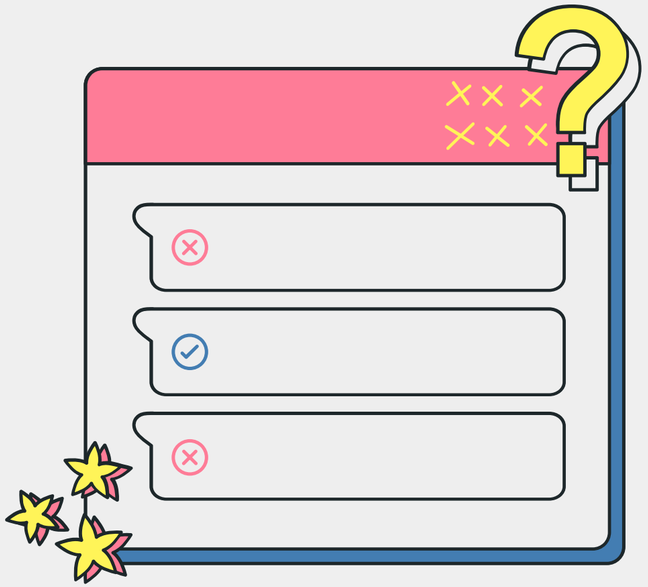
```
void agregarMiembro ( Persona *p ) {
  bool agregado = false;
  for ( int i = 0 ; i<5 ; i++ ){
    if (miembro[i] == NULL){
      miembro[i] = p;
      agregado = true;
    }
  }
  if (agregado == false){
    cout << "No hay cupos" << endl;
}
```

Si agregado sigue valiendo false, significa que no se pudo agregar.

Condicional IF FALSE

https://github.com/nicobrch/ayudantias-udp

Quiz



Quiz

1. Una clase puede tener múltiples métodos constructores. Digamos que la clase "Usuario" posee dos constructores en simultáneo, ¿Cómo puedo saber cuál usar en mi funcion main?
2. No puedo usar el condicional IF NULL si el arreglo no está inicializado, entonces ¿Cómo debo inicializarlo?
3. ¿Qué pasaría si modifico el tamaño del for a 10, pero no cambio el tamaño del arreglo?

Solucionario Quiz

1. Se puede distinguir entre constructores debidos a sus parámetros, es decir, los valores que van dentro del paréntesis. Si en mi función main uso "Usuario *u = new Usuario("Juan")", estaré usando el constructor que recibe un string, y no el vacío.
2. Para inicializarlo, se puede hacer un ciclo for dentro del método Constructor el cuál asigne los valores del arreglo a NULL.
3. Estaré accediendo a información la cuál no existe, lo cuál arrojará un error de memoria RAM u otorgará información equívoca.

https://github.com/nicobrch/ayudantias-udp

Ejercicios

Ejercicio 2

Un veterinario necesita que cree un programa con la clase "Gato", el cuál tiene nombre, color y edad. Además, deberá tener la clase "Veterinario", la cuál tiene una capacidad para atender 10 gatos a la vez. Cree este programa junto a los atributos necesarios y sus constructores. Además, cree la función "void agregarGato(Gato *g)", en la clase Veterinario, la agrega un gato a la capacidad del veterinario.

Ejercicio 3

Ahora, el veterinario tiene la capacidad de tener gatos en su veterinario, pero no la capacidad de finalizar su atención. Cree el método "finalizarAtencionGato(Gato *g)", el cuál busca dentro del arreglo a qué posición corresponde el gato y lo marca como NULL.

Solucionario Ejercicios

Los códigos se encuentran en canvas y en el siguiente link:

https://github.com/nicobrch/ayudantias-udp/tree/main/avanzada-2023-2