mylab

SUPER SHARING SPACE

POO, conceitos (cap 4)

Posted in programação (C plus plus) - 07/11/2018 - 0 Comment

as Classes:

#estruturas com funções membros
as funções que são declaradas como membros de uma estrutura são as **funções membro**
uma função membro sabe sempre para que objecto foi invocada
os membros privados só podem ser "trancados" se estes forem declarados como private, e desta forma apenas as funções membro conseguem aceder a eles
por omissão os membros de uma classe são privados, já se fosse os membros de uma estrutura são públicos.
qualquer função membro (nao estática) consegue aceder aos membros da classe sem referência explicita do objecto

acessibilidade:
se as funções membros forem as únicas a ter acesso directo à informação, então essas informações são membros privados
mantendo as funções membros como publicas, dentro da estrutura ou classe
as funções membros podem aceder a membros públicos ou privados, dentro da estrutura ou classe

para a aceder a um método de uma classe faz-se uso do operador ::

#construtores
serve para inicializar os objectos de uma classe
um construtor de uma classe pode ter parâmetros que podem representar os valores iniciais

#construtores overloaded
podemos ter vários construtores para especificar várias maneiras de inicializar objetos.
a escolha do construtor depende do dos argumentos que forem especificados
pode existir o construtor por omissão, será aquele que não tem especificado qualquer argumento

#construtores com parâmetros com valores por omissão
este tipo de construtor é também um construtor por omissão , pois pode ser chamado sem a especificação de argumentos

#destrutor
quando o objeto deixa de existir, os recursos devem ser libertados, usando destrutores, que são automaticamente invocados
a destruição dos objectos é feita da seguinte forma:
_objectos locais não estáticos são destruídos quando a execução do programa sai do bloco onde foram declarados
_objectos globais, são destruídos quando o programa termina
_objectos dinâmicos são destruídos recorrendo ao operador delete

#funções inline
são todas as funções que são funções membro de uma classe e que estejam definidas na classe
normalmente são funções pequenas e que são muitas vezes utilizadas

#membros variáveis estáticos (static)
é partilhado para todos os objectos de uma classe
existe apenas uma única existência de um e um só membro estático pelos objectos da classe
os não estáticos existe apenas uma existência por objectos
são definidos fora da classe, e pode ser inicializado

#funções estáticas (static)
têm acesso apenas aos membros estáticos (variáveis ou funções)
funções não estáticas tem acesso a todos os membros (estáticos ou não estáticos)
o acesso a um membro estático publico:
a.membro ou a.funçãoMembro()

o acesso a um membro estático publico sem referencia de objeto:
Disciplina::membro
Disciplina::funçãoMembro()

#funções membros constantes
o uso de const no fim do prototipo das funções indica que estas nao alteram os membros
se uma função membro tem const no final do prototipo também tem que aparecer const
as funções membros constantes podem ser invocadas para objectos constantes e nao constantes
as funções membros não constantes só podem ser invocadas para objectos não constantes

#funções friend
uma função friend relacionada com uma classe tem acesso aos seus membros privados não sendo mesmo membro dessa classe
a palavra friend surge antes do prototipo

#construtores com lista de inicialização
Disciplina():x(i), y(i){…
..
}

existem situações em que não é possivel inicializar membros variáveis dentro do construtor, tendo que se usar uma lista de inicialização
se forem membros constantes, têm que ser inicializados na lista de inicialização
se forem membros referencia, têm que ser inicializados na lista de inicialização

#Membros variáveis inicializados na declaração
é possivel inicializar os membros variáveis da classe aquando da declaração

#Membros default e delete
o uso da especificação default serve para indicar qual o construtor a ser utilizado (por omissão)
o uso da especificação delete serve para tornar indisponível a sua implementação

```
1
```

Tags : cap04, POO

Profile

Sign in with Twitter Sign in with Facebook
or

Comment

Name

- 97 Things Every Programmer Should Kno
- Building Skills in Programming
- Essential Coding Theory
- Foundations of Programming
- How to Design Programs
- How to Think Like a Computer Scientist
- The Codeless Code
- The Little Introduction To Programming

tools

- Download videos
- Rufos – iso to USB

Video Tutorials - making vide games

- curso: heartbeast [GM]
- Shaun Spalding – Video Tutorials
- Tom Francis – Video Tutorials

videojogos - blogs

- blog – Unseen64: Beta and Cancelled Videogames
- blog – Vida Extra

videojogos - conferencia

- Agenda eventos videojogos
- DevGAMM Conference
- IndieCade Europe
- International Conference on Entertainme Computing

videojogos - Devblogs

- Ganbatte Devblog

videojogos - framework

- Cocos2DX
- createjs
- flixel
- game froot
- Godot – opensource
- kiwijs
- Marmalade SDK
- phaser
- pico 8
- RPG Maker
- Torque
- wimi5