Processamento de Imagens

Turquesa - 2022/02

Aulas 7 e 8

Características da Linguagem C++

Linguagem C++

- Desenvolvida nos anos 80;
- É uma linguagem com suporte a orientação a objetos;
- Possui certa semelhança com a linguagem C, porém é muito mais poderosa; C e C++ são linguagem diferentes e independentes.

Entrada e saída e dados

- Também suporta as funções scanf() e printf(), porém, o ideal é utilizar as funções de entrada e saída de dados por fluxo:

Funções (necessitam da biblioteca iostream.h):

- **cin**: corresponde a entrada padrão
- **cout**: corresponde a saída padrão
- **cerr**: corresponde a saída padrão de erros

Entrada e saída e dados

- Exemplo de saída de dados:

```
#include <iostream.h>
void main(void)
{
 cout << “Olá mundo!\n“;
}
```

Entrada e saída e dados

- Exemplo de entrada de dados:

```
#include <iostream.h>
void main(void){
    char nome[50];
    cout << “Informe o seu nome: “;
    cin >> nome;
    cout << “Olá “ << nome << “, tudo bem?\n”;
}
```

Entrada e saída e dados

Vantagens dos fluxos de entrada e saída:

- **execução mais rápida:** a função printf() analisa a cadeia de formatação durante a execução do programa, enquanto os fluxos são traduzidos durante a compilação;
- **verificação de tipos:** como a tradução é feita em tempo de compilação, valores inesperados, devido a erros de conversão, jamais são exibidos;
- **código mais compacto:** apenas o código necessário é gerado; com printf(), porém, o compilador deve gerar o código correspondente a todos os formatos de impressão;

Tipos de Dados

Os tipos básicos de dados existentes no C++ são:

Tipo de Dado	Bits	Faixa de Valores
char	8	-128 a 127
bool	8	true ou false
int	32	-2.147.483.647 a 2.147.483.647
float	32	7 dígitos significativos
double	64	15 dígitos significativos

Conversão de tipos (cast)

Em C++, a conversão explícita de tipos pode ser feita tanto através da notação de cast quanto da notação funcional:

Exemplo:

```
int i, j;

double d = 3.54;

i = (int)d; // notação “cast”

j = int(d); // notação funcional
```

Tipos personalizados de dados no OpenCV

Tipos personalizados de dados - OpenCV

- A interface em C++ do OpenCV provê um tipo básico de estrutura para armazenar imagens: a classe Mat.
- Dependendo da forma como é criado, um objeto dessa classe é capaz de armazenar imagens (matrizes) de diversos tipos diferentes, tais como inteiros, floats, doubles, etc.

Tipos personalizados de dados - OpenCV

- Outros tipos também são predefinidos no OpenCV, tal como o tipo Vec3b.
- A classe Vec é definida no OpenCV para abrigar diversas formas de vetores curtos.
- A classe é definida por gabaritos e provê armazenamento de uma quantidade de valores de um dado tipo fornecido na instanciação do gabarito.

Tipos personalizados de dados - OpenCV

Exemplos:

- typedef Vec<uchar, 2> Vec2b;
- typedef Vec<uchar, 3> Vec3b;
- typedef Vec<uchar, 4> Vec4b;
- typedef Vec<int, 2> Vec2i;
- typedef Vec<int, 3> Vec3i;
- typedef Vec<int, 4> Vec4i;

**uchar => unsigned char*

Tipos personalizados de dados - OpenCV

Exemplo:

```
Vec3b val;
val[0] = 0;
val[1] = 0;
val[2] = 255;
```

Modelo de Cor utilizado no OpenCV

Modelo de Cor - OpenCV

O modelo de cor mais frequentemente utilizado para imagens é o RGB (Red, Green, Blue), porém o OpenCV adota um modelo de cor em que a ordem das cores é diferente, denominado BGR (Blue, Green, Red)

Exemplo com a configuração da cor vermelha:

```
Vec3b val;

val[0] = 0; //B

val[1] = 0; //G

val[2] = 255; //R
```

Fonte de consulta:

https://www.ime.usp.br/~slago/slago-C++.pdf

http://www.inf.ufpr.br/ci208/NotasAula.pdf