III.1.2. O C dentro do C++
Mais revisão de conteúdo – Linguagem C

- Regras de escopo
  - Definem onde uma variável é válida, onde foi criada e como ela sai do escopo (é destruída)
  - O escopo de uma variável vai do par "{}" mais próximo que a contém
    - Vai do ponto no qual ela foi definida até o "}" referente ao bloco ao qual pertence

```
// How variables are scoped
int main() {
    int scp1;
    // scp1 visible here
    {
        // scp1 still visible here
        //.....
        int scp2;
        // scp2 visible here
        //.....
        {
          // scp1 & scp2 still visible here
          //..
          int scp3;
          // scp1, scp2 & scp3 visible here
          // ...
        } // <-- scp3 destroyed here
        // scp3 not available here
        // scp1 & scp2 still visible here
        // ...
    } // <-- scp2 destroyed here
    // scp3 & scp2 not available here
    // scp1 still visible here
    //..
} // <-- scp1 destroyed here
```

III.1.2. O C dentro do C++
Mais revisão de conteúdo – Linguagem C

- Outros pontos importantes
  - Criação de tipos: ***typedef*** e ***structs***
    - ***Typedef***: possibilita uma descrição mais acurada de um tipo
      - Seu uso é importante junto a structs
      - Forma: `typedef existing-type-description alias-name`
      - Ex1: `typedef unsigned long ulong;`
      - Ex2:
```
int *x, y;          //x é ponteiro e y é int (* só vale para o 1°)
typedef int *IntPtr;    //define o tipo ponteiro para inteiro
IntPtr x, y;        //agora x e y são ponteiros
```
    - ***Struct***: forma de organizar um grupo de variáveis
      - Possui um conjunto de dados, denominados membros
        - Torna a operação de um conjunto de dados muito simples
      - Uma vez criada, possibilita gerar várias instâncias
        - Sua declaração não aloca espaço (apenas ao instanciar variável do tipo)
      - Formas: struct rotulo {tipo d1; … tipo dn;}; // só declara, pode instanciar várias
        struct {tipo d1; … tipo dn;} nome; // instancia 1 (nome) e não tem rótulo

III.1.2. O C dentro do C++

Mais revisão de conteúdo – Linguagem C

- Exemplo: `struct`

```
struct Structure1 {
  char c;
  int i;
  float f;
  double d;
};

int main() {
  Structure1 s1, s2;
  s1.c = 'a'; // Select an element using a '.'
  s1.i = 1;
  s1.f = 3.14;
  s1.d = 0.00093;
  s2.c = 'a';
  s2.i = 1;
  s2.f = 3.14;
  s2.d = 0.00093;
}
```

III.1.2. O C dentro do C++

Mais revisão de conteúdo – Linguagem C

- Outros pontos importantes
  - Criação de tipos: ***ponteiros*** e ***structs***
    - Às vezes é necessário manipular um objeto do tipo estrutura a partir de seu endereço, utilizando um ponteiro
      - Para obter seus elementos específicos a partir de um ponteiro, deve-se utilizar o operador '->'
      - Um ponteiro pode ser dinamicamente redirecionado para apontar para outro objeto de mesmo tipo

```
struct Structure3{
   char c;
   int i;
   float f;
   double d;
};

int main() {
   Structure3 s1, s2;
   Structure3* sp = &s1;
   sp->c = 'a';
   sp->i = 1;
   sp->f = 3.14;
   sp->d = 0.00093;
   sp = &s2; // Point to a different struct object
   sp->c = 'a';
   sp->i = 1;
   sp->f = 3.14;
   sp->d = 0.00093;
}
```

III.1.2. O C dentro do C++
Mais revisão de conteúdo – Linguagem C

- Outros pontos importantes
  - Criação de tipos: ***enum***
    - Tipo enumerado: é uma forma de atribuir nomes a números, propiciando maior legibilidade ao código
      - Forma: enum rotulo {nome1=v1, nome2=v2, ... Nomen=vn};
        - Se não houver atribuição de valor, nome1=0, nome2=1, ...
        - Podem ser atribuídos valores de referência, basta fazer a atribuição
          - Ex: enum ShapeType { circle = 10, square = 20, rectangle = 50};
        - Se para algum valor não houve atribuição, pega o inteiro seguinte ao valor anterior
          - Ex: enum exemplo { primeiro = 25, segundo };
            neste caso, segundo recebe valor 26
      - Nota: em C pode-se fazer operações com enum (ex: exemplo++)
        Em C++ já não pode (envolve 2 conversões de tipo enum ⇒ int ⇒ enum)

III.1.2. O C dentro do C++
Mais revisão de conteúdo – Linguagem C

- Outros pontos importantes
  - ***Vetores*** e ***Matrizes***
    - **Vetor** é uma estrutura unidimensional contendo elementos de um mesmo tipo
      - Forma: tipo nome[tam]; ➔ possui elementos de [0] a [tam-1]
        - Ex: `int a[10] = {10,20,30,40,50,60,70,80,90}` (a[9] recebe 0)
      - Notas:
        - Ao passar um vetor como parâmetro para uma função utilizando apenas o seu nome, na realidade está sendo passado seu endereço inicial de memória
        - Um vetor possui tamanho fixo, especificado na declaração
        - Em C++, é possível criar um vetor dinâmico utilizando a classe Vector
        - É possível ter um vetor de qualquer tipo, incluindo estruturas

```
struct ThreeDpoint{
  int i, j, k;
};

int main() {
   ThreeDpoint p[10];
   for(int i = 0; i < 10; i++) {
      p[i].i = i + 1;
      p[i].j = i + 2;
      p[i].k = i + 3;
   }
}
```

- **Matriz** é um vetor de vetores, ou seja, uma estrutura multidimensional contendo elementos de mesmo tipo
  - Sua manipulação é semelhante ao vetor, estendendo-o a várias dimensões

III.1.2. O C dentro do C++
Mais revisão de conteúdo – Linguagem C

- Diretivas de compilação
  - `#include`: inclui, na hora da compilação, um arquivo especificado
    - `#include "nomeArquivo"` : caminho relativo ao diretório de trabalho
    - `#include <nomeArquivo>` : caminho pré-especificados do compilador
  - `#define nomeMacro sequenciaCaracteres`
    - Diz ao compilador para substituir `nomeMacro` por `sequenciaCaracteres`
    - Deixa o programa mais geral (basta alterar no `#define`)
    - Exemplos de uso
      - `#define PI  3.1416` ou `#define VERSAO "2.02"`
      - `#define nomeMacro`: macro pode ser usada como uma espécie de *flag*
      - Simulando função no código, mas substituindo em tempo de compilação
        - `#define SQR(X) (X)*(X)` ou `#define ABS(a) ((a<0) ? (-a) : (a))`
        - `#define max(A,B)  ((A>B) ? (A):(B))` ou
          `#define min(A,B)  ((A<B) ? (A):(B))`
  - `#undef` : retira a definição (apagada da tabela interna que guarda as macros)