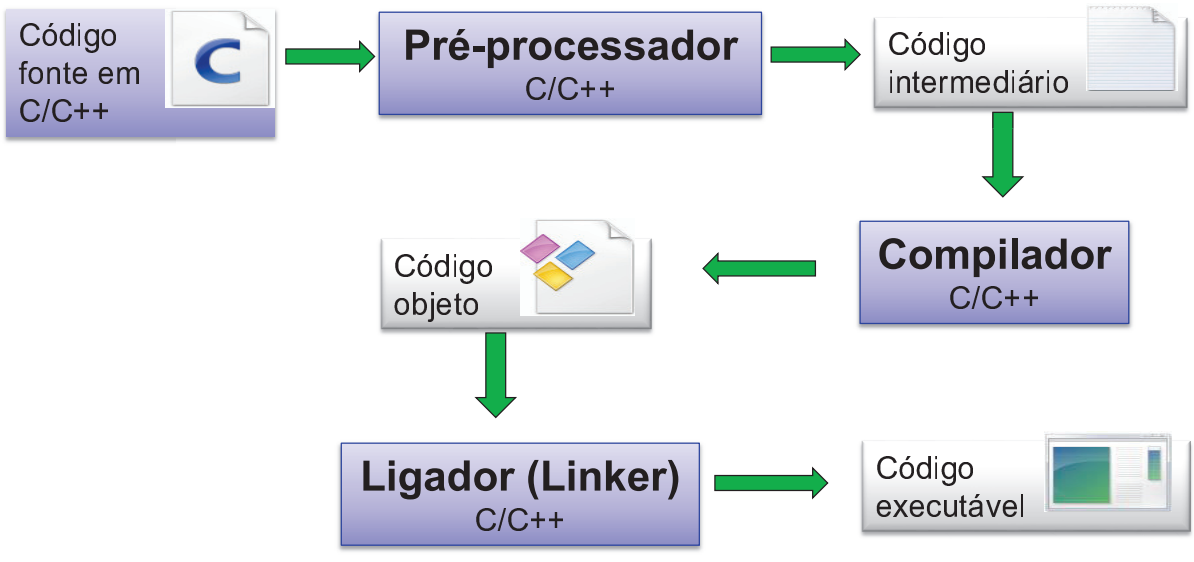
MTP – Métodos e Técnicas de Programação

Universidade Federal de Uberlândia
Prof. Renato Carrijo
renato.eletrica.ufu@gmail.com

Revisão

- Estrutura de um programa em C
- Variáveis e Constantes
- Laços
- Testes (if ... else ...)

Compilação de um programa em C



Identificadores

- São nomes usados para se fazer referência a variáveis, funções, labels e vários outros objetos definidos pelo usuário.
- O primeiro caracter deve ser uma letra ou um sublinhado.
- É case sensitive, isto é, as letras maiúsculas diferem das minúsculas.

Tipos de Dados

Tipo de dado	Significado	Tamanho (em bytes)	Intervalo de valores aceitos
char	Caractere	1	- de 128 a 127
unsigned char	Caractere não sinalizado	1	0 à 255
short int	Inteiro curto	2	-de 32 768 a 32 767
unsigned short int	Inteiro curto não sinalizado	2	de 0 a 65 535
int	Inteiro	2 (no processador de 16 bits) 4 (no processador de 32 bits)	-de 32 768 a 32 767 - de 2 147 483 648 a 2 147 483 647
unsigned int	Inteiro não sinalizado	2 (no processador de 16 bits) 4 (no processador de 32 bits)	de 0 a 65 535 de 0 a 4 294 967 295
long int	Inteiro longo	4	-de 2 147 483 648 a 2 147 483 647
unsigned long int	Inteiro longo não sinalizado	4	de 0 a 4 294 967 295
float	Flutuante (real)	4	$3.4*10^{-38}$ à $3.4*10^{38}$
double	Flutuante duplo	8	de $1.7*10^{-308}$ a $1.7*10^{308}$
long double	Flutuante duplo longo	10	$3.4*10^{-4932}$ à $3.4*10^{4932}$

Operadores

- ***Operador de atribuição***

O operador de atribuição em C é

o sinal de igual "=".

Operadores

- ***Operadores Aritméticos***

Os operadores *, /, + e - funcionam como na maioria das linguagens,

O operador % indica o resto de uma divisão inteira.

Operadores

- Operadores de Relação e Lógicos

- Relacionais

>	maior que
>=	maior ou igual
<	menor
<=	menor ou igual
==	igual
!=	não igual

- Lógicos

&&	and
||	ou
!	not

Operadores

- ***Incremento e decremento***

O C fornece operadores diferentes para incrementar variáveis. O operador soma 1 ao seu operando, e o decremento subtrai 1.

O aspecto não usual desta notação é que podem ser usado como operadores:

- ***pré-fixo*** (++x)
- ***pós-fixo*** (x++)

Operadores

- ***Operador cast***

Sintaxe:

(**tipo**) **expressão**

Exemplo:

```c
#include "stdio.h"
#include "conio.h"

void main()
{
    int i=1;
    printf("%d/4 e: %f",i,(float) i/4);
    getch();
}
```

Operadores

- ***Operador sizeof***

O operador **sizeof** retorna o tamanho em bytes da variável, ou seja, do tipo que está em seu operando.

Aplicação: Assegurar a portabilidade do programa.

Operadores

- ***Operador ternário***

Sintaxe:

condição?**expressão1**:**expressão2**

Aplicação: maneira compacta de expressar if-else.

Operadores

- ***Operador ternário***

Exemplo:

```c
#include "stdio.h"
#include "conio.h"

void main()
{
     int x,y,resultado;
     printf("Entre com dois números: ");
     scanf("%d",&x);
     scanf("%d",&y);
     resultado=(x>y)?1:0;
     printf("resultado= %d\n",resultado);
     getch();
}
```

Operadores

- ***Operador sizeof***

Exemplo:

```c
#include "stdio.h"
#include "conio.h"

void main()
{
     char ch;
     int i1;
     float f1;
     double d1;
     printf("Tamanho das variaveis: \n");
     printf("Char:     %d \n", sizeof(ch) );
     printf("Inteiro:  %d \n", sizeof(i1) );
     printf("Float:    %d \n", sizeof(f1) );
     printf("Doule:    %d \n", sizeof(d1) );
     getch();
}
```

Funções Básicas da Biblioteca C

Funções

- **printf**

Formatação:

\n	nova linha	%c	caractere simples
\t	tab	%d	decimal
\b	retrocesso	%e	notação científica
\"	aspas	%f	ponto flutuante
\\	barra	%o	octal
\f	salta formulário	%s	cadeia de caracteres
\0	nulo	%u	decimal sem sinal
		%x	hexadecimal

Funções

- **scanf**

Sintaxe:

scanf("expressão de controle", argumentos);

Exemplo:

```
#include "stdio.h"
#include "conio.h"


void main()
{
     int num;
     printf("Digite um número: ");
     scanf("%d",&num);
     printf("\no número digitado foi %d", num);
     getch();
}
```

Outras funções

- **getchar**

```
#include "stdio.h"

void main()
{
     char ch;
     ch=getchar();
     printf("%c\n",ch);
}
```

- **putchar**

```
#include "stdio.h"

void main()
{
     char ch;
     printf("digite uma letra: ");
     ch=getchar();
     putchar(ch);
     putchar('\n');
}
```

#define
e
type def

- *define*

```c
#define MAX 1000

int main() {
    int v[MAX];
    int i;
    for (i = 0; i < MAX; ++i) {

    }

}
```

- *typedef*

```c
#include <stdio.h>
#include <string.h>

typedef struct Books
{
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
} Book;

int main( )
{
   Book book;

   strcpy( book.title, "C Programming");
   strcpy( book.author, "Nuha Ali");
   strcpy( book.subject, "C Programming Tutorial");
   book.book_id = 6495407;

   printf( "Book title : %s\n", book.title);
   printf( "Book author : %s\n", book.author);
   printf( "Book subject : %s\n", book.subject);
   printf( "Book book_id : %d\n", book.book_id);

   return 0;
}
```

Exercícios

Exercício 1

Faça um programa que efetue o cálculo das raízes de uma equação do segundo grau.
A equação será na forma $ax^2 + bx + c = 0$, e o usuário deverá fornecer os coeficientes ***a***, ***b***, e ***c***.
Considere:
- O valor do coeficiente ***a*** deve ser diferente de zero, caso contrário, o programa deverá exibir a mensagem "Esta não é uma equação de segundo grau".
- Se Delta < 0, então não existem raízes reais para a equação. Imprimir a mensagem "Não existe raiz real".
- Se Delta = 0, então existe apenas uma raíz real. Imprimir a mensagem "A raíz encontrada é raíz única".
- Se Delta > 0, então existem duas raízes reais. Imprimir os valores das raízes.

Obs.: Usar uma função ***calculaRaizEQ2Grau*** .

Restrições: o código interno da função ***calculaRaizEQ2Grau*** deverá conter apenas operações relativas aos cálculos. Não utilizar dentro da função as operações scanf, printf.

Exercício 2

Faça um programa que leia um vetor de números inteiros de 10 posições. O programa deve calcular e mostrar:

a. O maior elemento do vetor e em que posição esse elemento se encontra;
b. O menor elemento do vetor e em que posição o elemento se encontra.

Exercício 3

Escreva uma função que receba por parâmetro um valor inteiro e positivo **n** e retorne o valor da soma **s** dada a seguir:

$$s = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{n!}$$

Você também deverá fazer a função *fatorial* (que devolva o fatorial de um dado número inteiro) e utilizá-la na implementação da função para o cálculo da soma anterior.