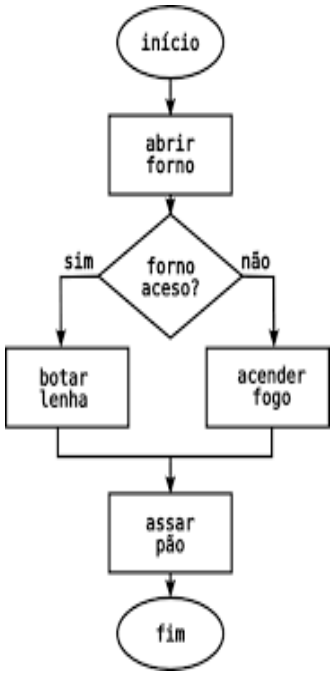
Técnicas de Desenvolvimento de Algoritmos



- Introdução a linguagem C;

CLÓVIS JOSÉ RAMOS FERRARO

cferraro@unicid.edu.br

Exercício 1

Quatro suspeitos de praticar um crime fazem as seguintes declarações:

João: Carlos é o criminoso

Pedro: eu não sou criminoso

Carlos: Paulo é o criminoso

Paulo: Carlos está mentindo

Sabendo que apenas um dos suspeitos mente, determine quem é o criminoso.

a) João
b) Pedro
c) Carlos
d) Paulo

Exercício 2

(Vunesp/TJ-SP) Sabendo que é verdadeira a afirmação “Todos os alunos de Fulano foram aprovados no concurso”, então é necessariamente verdade:

a) Fulano não foi aprovado no concurso.
b) Se Roberto não é aluno de Fulano, então ele não foi aprovado no concurso.
c) Fulano foi aprovado no concurso.
d) Se Carlos não foi aprovado no concurso, então ele não é aluno de Fulano.
e) Se Elvis foi aprovado no concurso, então ele é aluno de Fulano.

Exercício 3

(FGV/ TJ-AM) Dona Maria tem quatro filhos: Francisco, Paulo, Raimundo e Sebastião. A esse respeito, sabe-se que:

I. Sebastião é mais velho que Raimundo.
II. Francisco é mais novo que Paulo.
III. Paulo é mais velho que Raimundo.

Assim, é obrigatoriamente verdadeiro que:

a) Paulo é o mais velho.
b) Raimundo é o mais novo.
c) Francisco é o mais novo.
d) Raimundo não é o mais novo.
e) Sebastião não é o mais novo.

Meu primeiro programa em C

```
#include <stdio.h>
void main ()    /* Um Primeiro Programa */
{
     printf ("Ola! Meu primeiro programa!\n");
 }
```

O que vamos usar?

Ferramenta On-line:

https://www.onlinegdb.com/

https://replit.com/

https://www.programiz.com/c-programming/online-compiler/

CODEBLOCK:

https://sourceforge.net/projects/codeblocks/files/Binaries/20.03/Windows/codeblocks-20.03mingw-setup.exe/download

Meu primeiro programa em C

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

A linha **#include <stdio.h>** diz ao compilador que ele deve incluir o arquivo-cabeçalho **stdio.h**. Neste arquivo existem declarações de funções úteis para entrada e saída de dados (std = standard, padrão em inglês; io = Input/Output, entrada e saída ==> stdio = Entrada e saída padronizadas).

ARQUIVOS DE CABEÇALHO PADRÃO ANSI C

ARQUIVO	FINALIDADE
ctype.h	Declara funções para testar caracteres
math.h	Define as funções matemáticas e macros usadas para operações matemáticas em linguagem C
stdlib.h	Contém funções para conversão de números, alocação de memória e tarefas similares
string.h	Define funções para manipulação de “strings”
time.h	Define funções e estruturas usadas na manipulação de data e hora.

Meu primeiro programa em C

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

A linha **void main()** define uma função de nome **main**. Todos os programas em C têm que ter uma função **main**, pois é esta função que será chamada quando o programa for executado. O conteúdo da função é delimitado por chaves **{ }**.

Meu primeiro programa em C

```
#include <stdio.h>
int main ()    /* Um Primeiro Programa */
{
    printf ("Ola! Meu primeiro programa!\n");
    return 0;
 }
```

Identificadores

São os nomes que podem ser dados para variáveis e funções.
Para a escolha destes nomes é necessário seguir algumas regras:

- Um identificador deve iniciar por uma letra ou por um "_" (*underscore*);
- A partir do segundo caracter pode conter letras, números e *underscore*;
- Deve-se usar nomes significativos dentro do contexto do programa;
- C é uma linguagem *case-sensitive*, ou seja, faz diferença entre nomes com letras maiúsculas e nomes com letras minúsculas.
  - Peso e peso são diferentes;
- Costuma-se usar maiúsculas e minúsculas para separar palavras: "PesoDoCarro";
- Deve ser diferente dos comandos da linguagem (palavras reservadas);
- Deve ter no máximo 31 caracteres (no caso do TurboC);
- Pode conter números a partir do segundo caracter;

Tipos de Variáveis

- Todas as variáveis em C tem um ***tipo***;
  - Cada tipo define os valores que a variável pode armazenar;
  - Cada tipo ocupa uma certa quantidade de memória.

Tipo	**Valores Válidos**
char	letras e símbolos: 'a', 'b', 'H', '^', '*','1','0'
int	de -32767 até 32767 (apenas números inteiros)
float	de -3.4×10^{38} até $+3.4 \times 10^{+38}$com até 6 dígitos de precisão
double	de -1.7×10^{308} até $+1.7 \times 10^{+308}$com até 10 dígitos de precisão

Tipos de Variáveis

Tipo	Num de bits	Intervalo	
		Inicio	Fim
char	8	-128	127
unsigned char	8	0	255
signed char	8	-128	127
int	16	-32.768	32.767
unsigned int	16	0	65.535
signed int	16	-32.768	32.767
short int	16	-32.768	32.767
unsigned short int	16	0	65.535
signed short int	16	-32.768	32.767
long int	32	-2.147.483.648	2.147.483.647
signed long int	32	-2.147.483.648	2.147.483.647
unsigned long int	32	0	4.294.967.295
float	32	3,4E-38	3.4E+38
double	64	1,7E-308	1,7E+308
long double	80	3,4E-4932	3,4E+4932

Inicialização de Variáveis

```
char ch, letra;
long contador;
float pi;
```

Inicialização de Variáveis

```
{
  int NroDeHoras = 0; // declara e inicializa com Zero
  float PrecoDoQuilo = 10.53;
  double TaxaDoDolar = 1.8,
  TaxaDoMarco = 1.956,
  TaxaDoPeso = 1.75;
  .......
}
```

Constantes

Constantes são identificadores que não podem ter seus valores alterados durante a execução do programa.
Para criar uma constante devemos utilizar o comando #define que, em geral é colocado no início do programa-fonte.

Exemplos

```
// Não se coloca ponto-e-vírgula após o valor
#define PI 3.1415
#define UNIVERSIDADE “Universidade Cidade de São Paulo - UNICID“ // deve-se colocar entre aspas
```

A função *printf*

A função printf exibe um ou mais dados na tela. Para tanto ele deve receber pelo menos dois parâmetros, separados por vírgula:

Por exemplo:
printf("%s","teste");

"%s" : é a string de formato
"teste" : é o dado a ser impresso.

```
void main ()
{
printf("%s","Isto é uma string ....\n"); // note o '\n' no final da string;
printf("%s %s","teste1", "outra string\n");
printf("A aluna %s ficou doente","Maria");
}
```

Alguns Especificadores de Formato da Função PRINTF()

Especificador	Tipo do Argumento	Descrição
%d	int	Valor inteiro decimal
%i	int	Valor inteiro decimal
%c	char	Um caracter em formato ASCII (código binário correspondente)
%s	string	Uma cadeia de caracteres (string)
%f	float	Valor em ponto flutuante
%e	float ou double	Valor em ponto flutuante em notação exponencial

A função *printf - Inteiros*

```
#include <stdio.h>
#include <conio.h> // necessário para as funções clrscr e getchvoid
#include <stdlib.h> // necessário para as funções system(“cls”) e (“pause”)
void main () {
int NroDeAndares, AlturaPorAndar ;

//  clrscr(); // Limpa a tela
// system("clear"); // Limpa a tela no windows
// system("cls"); // Limpa a tela no linux

NroDeAndares = 7;
AlturaPorAndar = 3;

printf(“O total de andares no predio eh: %d\n", NroDeAndares);

printf("Altura Total do Prédio: %d metros", NroDeAndares*AlturaPorAndar);

//getch(); // espera que o usuário pressione uma tecla
//System(“pause) // espera que o usuário pressione uma tecla
}
```

A função *printf – Números reais*

```
float Numero;
Numero = -2.5;
clrscr();
printf("1234567890\n");
printf("%7f\n", Numero);
printf("%7.0f\n", Numero);
printf("%7.3f\n", Numero);
printf("%8.3f\n", Numero);
printf("%9.3f\n", Numero);
printf("\n");
printf("%8.4f\n", Numero);
printf("%8.1f\n", Numero);
printf("%6.12f\n", Numero);
printf("%-7.3f\n", Numero)
```

CARACTERES DE ESCAPE

Código	Significado
\a	Sinal sonoro ("beep")
\b	Retrocesso (“backspace”)
\n	Mudança de linha (“new line”)
\f	Avanço de página (“form feed”)
\r	Retorno do carro da impressora
\t	Tabulação horizontal (“tab”)
\v	Tabulação vertical
\\	Impressão da barra invertida
\”	Impressão das aspas
\’	Impressão do apóstrofo
\?	Impressão do ponto de interrogação

Operadores matemáticos

-	sinal de menos (unário)	Maior precedência (avaliado antes)
*,/	multiplicação e divisão	
%	módulo (resto da divisão)	
+,-	soma e subtração	Menor precedência (avaliado depois)

A função *scanf*

A função ***scanf*** é responsável por ler um dado do teclado e imediatamente atribuí-lo a uma variável; . Para tanto esta deve receber pelo menos dois parâmetros, separados por vírgula:

Por exemplo:
scanf("%s",&teste);

"%s" : é a string de formato
“&teste" : é a variável que irá receber o dado.

```
void main ()
{
printf("%s","Isto é uma string ....\n"); // note o '\n' no final da string;
printf("%s %s","teste1", "outra string\n");
printf("A aluna %s ficou doente","Maria");
}
```

A função scanf

```
#include <stdio.h>
void main ()        {
          int Dias;/* Declaracao de Variaveis*/
          float Anos;
          printf ("Entre com o número de dias: ");
          /* Entrada de Dados */
          scanf ("%d",&Dias);
         Anos=Dias/365.25;
         /* Conversao Dias->Anos */
        printf ("\n\n%d dias equivalem a %f
anos.\n",Dias,Anos);        }
```

Exercício

- O que faz o seguinte programa?

```
#include <stdio.h>
main()
{
    int x;
     scanf("%d",&x);
     printf("%d",x);
}
```

Exercício

- Desenvolva uma rotina que solicite ao usuário que informe o número de andares de um prédio, a altura padrão dos andares. A rotina deve calcular e exibir a altura total do prédio.

```
#include <stdio.h>
#include <stdlib.h>
void main () {
int NroDeAndares;
float Altura, ALturaTotal;
NroDeAndares = 0;
AlturaPorAndar = 0.0;
printf("Informe o número de andares do predio eh:\t");
scanf("%i",&NroDeAndares);
printf("\nInforme a altura padrão de um andar\t");
scanf("%f",&Altura);    ALturaTotal=NroDeAndares*Altura;
printf("\nA altura total do predio eh: %f
m",NroDeAndares*Altura);}
```

Variáveis do Tipo String

- Uma variável capaz de arrmazenar uma string deve ser declarada informando-se qual o número máximo de caracteres que ela poderá armazenar.
- **Exemplo:**
  char Nome[30]; // isto define que a variável poderá armazenar uma
  // string de até **29** caracteres**.**
- Quando trabalhamos com strings deve-se incluir o arquivo de cabeçalho ***string.h***

Atribuição com strings

```
#include <stdio.h>
#include <conio.h>
#include <string.h> // arquivo de cabeçalho para trabalhar com strings

void main() {
char Nome[30]; // declara uma string que poderá armazenar até 29 caracteres !!
clsrscr();
strcpy(Nome, "Jose da Silva"); // atribui "Jose da Silva" para a variável Nome
printf("O funcionário %s foi tranferido", Nome); // no lugar de %s aparecerá o // conteúdo da variável Nome
getch(); }
```

Operadores matemáticos

Símbolo	Operação	Exemplo	Descrição
+	Adição	a + b	Soma os dois operandos
-	Subtração	a - b	Subtrai o segundo operando do primeiro
*	Multiplicação	a * b	Multiplica os dois operandos
/	Divisão	a / b	Divide o primeiro operando pelo segundo
%	Módulo	a % b	Divide o primeiro operando pelo segundo operando e produz a parte restante

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int i, j, k;
    clrscr();
    printf(“Informe o valor  de i: ");
    scanf("%d", &i);
    printf(" Informe o valor  de j: ");
    scanf("%d", &j);
    k = i + j;
    clrscr();
    printf(“O valor de i e %d\n", i);
    printf(“O valor de j e %d\n", j);
    printf(“O valor da somade i e j e %d\n", k);
    getch();
    clrscr();
    return 0;}
```

Exibindo caracteres especiais no printf.

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
void main () {
Setlocale(LC_ALL, “Portuguese”);
printf("%s",“Meu nome é José\n");
}
```