Curso de Verão: Introdução à Programação

04.01.2018

Dados p/ acesso (wi-fi): Nome: verao2018
Senha: sempreusesoftwarelivre

Monitoria: Débora - Terças e sextas (10:00 às 11:00)
- Sala de monitoria - Bloco B
  - PACA - Dúvidas online
  - Palavra-chave: IMEX8

emails: -> debora.buss@hotmail.com
-> macielcalebe@gmail.com

Juiz Online: URI online Judge

Avaliação: Listas do URI
- Listas de exercícios
- 2 Provas
- Provinhas (10 ~ 15 minutos)

Média: MF: ((MP + MI) . 0,8) + (ML . 0,2)
- MP: Média Provas
- MI: Média Provinhas
- ML: Média Listas

Caso (MP + MI) < 50,00 : MF = (MP + MI)

Frequência: F ≥ 85%

- Certificados à partir de agosto/2018.

Primeira separação:

- Hardware: parte física.
- Software: parte lógica.

05.01.2018

- Processador (cérebro):
  - Un. Controle
  - Un. Lógica e aritm.
    - Regist. (Mem. de trabalho)
- Memória (Arm. dados)
  - Mem. RAM
    - Caches (Devolver rápido regiões)
  - Mem. HD (Disco) (Memória de inativos)

- Barramentos: Linhas de comunicação.
- Unidade de entrada e saída (IO):
  - → Saída: Caixas de som, monitor...
  - → Entrada: Microfone, teclado...

- Sistema operacional (S.O.):
  - → Abstração de Hardware
  - → Intermédio entre programas e componentes físicos

Algoritmo

- É uma sequência finita de instruções bem definidas e não ambíguas.
- Passos para realizar uma tarefa
- Não representa necessariamente um programa.

1. Algoritmo para ir trabalhar pela manhã:
   1- Acordar
   2- Tomar banho
   3- Tomar café da manhã
   4- Escovar os dentes
   5- Pegar carteira
   6- Pegar chave de casa
   7- Sair de casa
   8- Trancar a porta
   9- Ir até a estação
   10- Pagar o trem
   11- Pegar o trem
   12- Sair na estação
   13- Ir ao trabalho

→ Características:
- Finitude -> nº de passos, tempo e recursos.
- Exatidão -> definição de passos.
- Efetividade -> cumprir seu propósito.
- Entradas
- Saídas

2. Converter polegadas para centímetros

Alg. conv. pol => cent. (POLCENT)

Entradas: Medida em pol, fator de conv.

1. Devolva a Medida em pol multiplicada pelo fator de conv.

POLCENT (2, 2.54) => 5,08 (fator de conv. arbitrário)

Relembrando:

1. Devolva a medida em pol. multiplicando * 2.54

08.01.2018

3, 5, 8, 12, 15

3. Dada uma lista de idades de pessoas, informar a idade da pessoa mais velha.

Entrada: lista de idades

1- Percorra em ordem crescente (completo)
2- Retorne o último valor

5- Defina a 1ª idade como MAX.
6- Do 2º até a última idade, compare com MAX e troque quando for maior.
8- Devolva MAX.

-> Gambiarra: definir MAX como 0.

4. Responder se existe, no estoque de uma empresa, um parafuso com certa medida.

Entrada: lista com as medidas dos parafusos

1- Comparar X com cada medida da lista. Se for igual, retorna SIM
2- Devolva NÃO

Linguagem

- Oquê é uma linguagem?

Um código para se comunicar.

Linguagem de programação:

"... é um método padronizado p/ se comunicar instruções para um computador."

- Permite especificar:

Linguagem interpretada x Compilada

- Trad. do cód. fonte p/ cód. de máq.

- Interpretada
  - O programa é executado conforme vai sendo traduzido.

- Compilada
  - Todo o cód. fonte é traduzido antes de ser executado.

Paradigma

- Estrutural
  - Sequência, decisão e iteração
  - Estruturas, como sub-rotinas e funções
  - Resolver prob. mais simples e diretos
  - C, Cobol, Pascal e Perl.

- Orientada a objetos
  - Modelar o mundo real no domínio do prob.

-> Linguagens de Alto e Baixo nível.

- B. nível
  - L. de máquina
  - 0s e 1s

- A. nível
  - nota final = nota2 + notaextra

Assembly
load nota
add notaextra
store notafinal

C é uma das linguagens de programação mais próximas ao baixo nível

- C
  - -> L. compilada
  - -> Desenvolver sistemas operacionais
    - Unix
    - Quase o B. nível de hardware.
  - -> Padrão
    - ANSI C

Preparar-se!

- Programas necessários:
  - Editor de código
  - Compilador

-> Notepad++ -> MinGW
-> Komodo edit
-> Bloco de notas

Dar ênfase sobre o diretório

Programando...

-> buscar a biblioteca: #include <stdi

* Antes de todos programas escrevemos <stdio.h> a biblioteca.

```c
#include <stdio.h>
int main() {
    return 0;
}
```

→ Uma prog não faz nada.
main é onde região (escopo)

```c
#include <stdio.h>

main() {
    printf("Olá IMG");
    return 0;
}
```

Em C o main é executado antes que tudo, independente da ordem

\n é quebra de linha (enter) entre as aspas

// → comentários

→ Como usar o prompt de comando?

dir (diretório) aparece todos nesta acima. -eis -

cd.. → Volta um ↑ na hierarquia. Tem sempre que obedecer a hierarquia

cd Dropb

(Linha em branco: Dev c

Bit: 0 ou 1.

Byte: conjunto de 8 bits.

1 variável pelo menos 1 byte

→ Tipo e identificadores logo depois do main (nesta ordem)

- float: nº real
- int: nº inteiro

```c
#include <stdio.h>
// (comentário
```

Compilação: gcc -Wall -ansi -pedantic-errors prog.c -o prog.exe

dir *palavra → Cria palavra

dir i*.c → começa com i e tem .c em C.

10.01.2018

• Construa um programa que tenha info. sobre idade e salário de um usuário.

```c
/* Crie msg c/ a idade e salário de alguém */

#include <stdio.h>

int main() {

    int idade;
    float salario;

    idade = 23;
    salario = 2000.00;

    printf("Você tem %d anos e seu salário é R$ %f\n", idade, salario);

    return 0;
}
```

Identificadores: identifica um endereço de memória. Use-se: A-z; 0-9; _. Mas um qual em minúsculo e não pode começar com nº. # variáveis.

- %d → retorno inteiro
- %f → retorno real

```c
/* Converte polegadas em pé centímetros */
#include <stdio.h>
int main () {

    float polegadas, cm;

    polegadas = 55
    cm = polegadas * 2.54

    printf ("%f polegadas equivalem à %f centímetros.", polegadas, cm);

    return 0;
}
```

Entradas:

5.4: Substitui por scanf("%f", &polegadas);

- scanf ("<tipo>", &variável)
- %.3f → 3 casas decimais

- scanf("%f %f", &valor1, &valor2)
Se separado por vírgula:
scanf("%f, %f", &, &)
...

11.01.2018

```c
/* Calcular a área de um retangulo */

#include <stdio.h>

int main() {
    float base, altura;

    printf("Informe o valor da base: \n");
    scanf("%f", &base);

    printf("Informe o valor da altura: \n");
    scanf("%f", &altura);

    printf("A area do retangulo eh: %.3f\n",
    base*altura);

    return 0;
}
```

```c
/*Crie um prog. que calcule para o usuário o
consumo em uma viagem */

#include <stdio.h>

int main(){
    float km_por_litro, preco_litro, Km;
    float valor_total, qtde_litros;

    printf("Quantos Km/L percorre?");
    scanf("%f", &km_por_litro);

    printf("Preço do combustivel por litro");
    scanf("%f", &preco_por_litro);

    printf("Quantos Km vai percorrer?");
    scanf("%f", &Km);
    qtde_litros = Km / km_por_litro;
    printf("Vc precisa abastecer %.2f litros\n",
    qtde_litros);
    printf("O custo total é %.2f reais\n",
    Km * preco_litro);

    return 0;
}
```

Palavras

- Variáveis: Os nomes de variáveis não podem ser iguais a palavras-chaves.

- Identificadores:
  - → Convenções:
    - lowercase
    - lower_case_with_underscores
    - mixedCase
    - UPPERCASE
    - UPPER_CASE_WITH_UNDERSCORES

* P/ constantes!

- Tipagem:
  - → Forte: Sempre tem um tipo de variável associado.
  - → Estática: As variáveis enquanto existem sempre tem o mesmo tipo.

- Declarar variáveis: → mostrar o tipo no código main.
- Inicializar variáveis: → determinar variáveis.

- Tipos:
  - int → pos e neg.
  - float
  - double
  - char
  - void

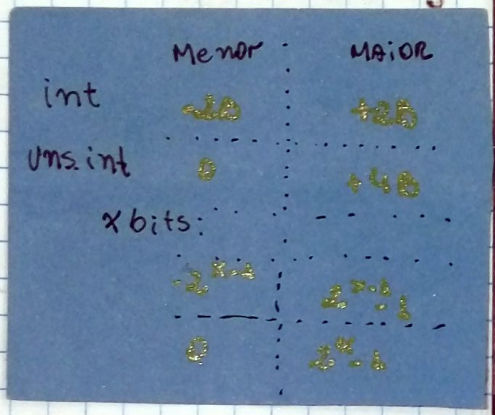
  - → Int: F. ent.: %d, ou %i.
    F. saída: %d, %l, %o, %x, ou %X.
  - → float: F. ent: %f
    F. saída: %f, %e, ou %g → ainda
  - → double:
    F. ent: %lf
    F. saída: %f, %e, ou %g.
  - → char: F: %c (-128 a 127)

```c
/* Int do tab ASCII p/ um char */
#include <stdio.h>
int main(){
    char letra;

    printf("Informe uma letra qualquer: ");
    scanf("%c", &letra);

    printf("\n Letra informada: %c\n", letra);
    printf("\n Posição na tabela: %d\n", letra);

    return 0;
}
```

15.01.2018

```c
#include <stdio.h>

int main(){

    char letra, letraMaiuscula;

    printf("Informe uma letra minuscula: ");
    scanf("%c", &letra);

    letraMaiuscula = letra - 32;

    printf("A letra maiuscula e': %c\n",
    letraMaiuscula);

    return 0;
}
```

- **Modificadores**
  - unsigned - sem sinal
  - signed - com sinal
  - long
  - short



- signed int num = int num
- unsigned int num => 4 bytes positivos. Ex: %u
- short int num => 2 bytes pos. e neg.
- long int num => 8 bytes pos. e neg. Ex: %Ld

- **Constantes**: UPPERCASE
  - UPPER_CASE_WITH_UNDERSCORES

Trocar a constante p/ leitura: #define ou const.

```c
/* Area do circ. utilizando defines */

#include <stdio.h>

int main(){
    double raio;
    double const PI = 3.1415;

    printf("Informe o raio: ");
    scanf("%lf", &raio);

    area = raio * raio * PI;
    return 0;
}
```

- #define ANTES do int main!!!
- #define entrada do scanf => SUBSTITUI!!! :)
  #define saida printf

15.01.2018

Operadores aritméticos

- + - * / (ad., sub., mult., div.)
- % (Resto da div. inteira)
- Potenciação por enquanto manual!!! (sem operador :( )

Type casting

=> Transformar o formato da saída:

- Ex: res = int1/int2 = int
  res = (double) int1/int2 = double

A divisão de dois int irá retornar um int

Usar ponto p/ sair float ou double:
=> 5/2 -> 2,0000
=> 5,0/2 -> 2,50000

Prova

Atribuição + operador aritmético:

- qtde += 1 = qtde = qtde + 1
- total -= desc = total = total - desc
- res *= 3 = res = res * 3
- valor %= 2 = valor = valor % 2

Limitar casas no printf

=> Utilizar %8.2f como exemplo:
Limita um número que utiliza oito colunas na tela e exibe apenas três casas decimais.
=> Imprimir zeros nos espaços vazios:
Ex: %08.2f

If e Else

- Usar chaves { } nos blocos de if e else.

```
...
    if (media >= 70.0) {
        printf("Aprovado\n");
    } else {
        printf("Reprovado\n");
    }...
```

- Comparação:
  - Igual a
  - Diferente de (comparação de tipos, não de tipos)

Estruturas de seleção

Ex: 5.6 == 5.60

- Zero (0) p/ falso
- Um (1) p/ verdadeiro

Operadores lógicos

- Negação: !
- e: &&
- ou: ||

T	T	T	T	T	T
T	F	T	T	F	F
F	T	F	F	T	F
F	F	F	F	F	F

Ordem: ! → && → ||

- Comparação com letra:
  Ex: sexo == 'F'
- Não é necessário ter bloco else no bloco if.

18.01.2018

- Dá para comparar letras sem char:
  letra <= 'g' só exposições simples
- Qualquer modelo de if else é correto.

19.01.2018

Operador Ternário

```
1  /* Lê dois ints e guarda o maior */
2  #include <stdio.h>
3
4  int main (){
5
6      int num1, num2, maior;
7
8      printf("Informe dois int: ");
9      scanf("%d %d", &num1, &num2);
10
11     maior = (num1 >= num2)? num1 : num2;
12
13     return 0;
14
15 }
```

%s é o formatador para uso de palavras

Ex: printf("O int %d eh %s\n", numero, (numero%2 == 0)? "par" : "impar");

Entre aspas

Switch

```
1  /* Conceito c/switch */
2
3  #include <stdio.h>
4
5  int main (){
6
7      char conceito;
8      printf("Informe o conceito [A,B,C,D,R]: ");
9      scanf("%c", &conceito);
10
11     switch (conceito){
12         case 'A': printf("Excelente\n");
13         case 'B':
14         case 'C':
15         case 'D':
16         case 'R':
17         default: printf("Conceito invalido!\n");
18     }
19 }
```

Revisar

break !!!

* switch só funciona p/ ints !

Estruturas de Repetição

While

```
while (<condição>)
{
    <comandos>   T
    <comandos>   T   quando
    <comandos>   T     F pula para
}                      fora do while

<comandos para dow>
<comandos fora do w>
```

```
/* Repetir msg no título */
#include <stdio.h>

int main (){

    int qtde;

    qtde =0;

    while (qtde <= 3){
        printf ("Se o número for par\n");
        qtde = qtde + 1;
    }

return 0;
}
```

27/01/2018

```
/* Validação de valor em [0,100] */

#include <stdio.h>

int main (){
    double nota;
    scanf ("%lf", &nota);
    while (nota<0.0 || nota>100.00){
        printf("Nota inválida! Digite novamente\n");
        scanf("%lf", &nota);
    }
    printf ("\n A nota foi %.2f\n", nota);

return 0;
}
```

Substituir as linhas 9 a 12 por:

```
1  int main (){
2      double nota;
3      do {
4          printf("Informe a nota [0,100]:");
5          scanf("%lf", &nota);
6          if (nota<0.0 || nota>100.0) {
7              printf("Nota inválida!\n");
8          }
9          while (nota >=0.0 || nota<=100.0){
10             printf("Nota %.2f\n",nota);
11         }
12
13
14
15
16
17
18
19
20
```

For

```
for ([expInit]; [expControl]; [expInd])
```

```
/* Exemplos com for */

#include<stdio.h>

int main() {
    int num;

    for (num=1; num<=10; num=num+1) {
        printf("Hoje é segunda\n");
    }

    return 0;

}
```

```
/* Encontrar o maior valor digitado c/ o for */

#include <stdio.h>

int main(){
    int num, maior, n, i;

    printf("Quantos valores você vai informar?\n");
    scanf("%d",&n);

    for(i=1; maior=0; i<=n; i=i+1)
        /*printf("%d\n", i);*/ -> p/ visualizar como o for está rodando
        printf("");
        scanf("%d", &num);
```

```c
        if(num > maior) {
            maior = num;
        }
    }
    printf("O maior valor eh: %d\n", maior);

    return 0;
}
```

Repetição com quantidade definida: for

Repetição sem quantidade definida: while

* Estudar o código faixaIdade.c no PACA.

!!! Lixo de memória: Tomar cuidado com variáveis não inicializadas !!!

* Simplificar as relações de comparação -> Não ser repetitivo!

-> Média de P1 e P2 entre 0 e 10!

!!! Na prova não se pode usar a biblioteca string.h !!!

* Exemplo de potência com est. de repetição

```c
/* Potencia */

#include <stdio.h>

int main() {
    int base, exp, i, res;

    printf("Informe a base: ");
    scanf("%d", &base);

    printf("Informe a exp: ");
    scanf("%d", &exp);

    for(i=1, res=1; i<=exp; i++)
    {
        res = res * base;
    }
    printf("\n%d elevado %d eh igual a %d\n",
        base, exp, res);

    return 0;
}
```

Strings

-> char msg[50] = "Msg c/ até 50 posições!";

printf("A msg eh: %s\n", msg);

-> msg[0] = '@' -> Substitui um termo na posição 0 (primeiro) por @.

Leituras:

V/ ser lido não usa-se "&"

```c
char cantor[50];
scanf("%s", cantor);
```

↳ Tira o &!!!

↳ ou mete, & cantor[0]

* para usar com espaços: %[^\n] no scanf

-> fgets lê com o \n.

↳ standart input

* fgets(cantor, 50, stdin);

Contando caracteres de um texto:

```c
/* Conta o tam de um texto */

#include <stdio.h>

int main() {
    char texto[100];
    int i, tam;

    printf("Informe um texto: ");
    scanf("%[^\n]", texto[0]);
    tam = 0;
    for(i=0; texto[i] != '\0'; i=i+1) {

        /*printf("%c\n", texto[i]);*/
        tam = tam + 1;
    }

    printf("O txt informado foi: %s", texto);
    printf("O tamanho do txt e: %d\n", tam);

    return 0;
}
```

```
for (tam=0; texto[tam]!='\0'; tam=tam+1);
```

é um FOR já printf("O texto... é o mesmo programa que o anterior !!!

P/ contar os espaços em um texto:

```
for( =0; qtde=0; texto[i]!='\0'; +=1){
    if(texto[i]==' '){
        qtde=qtde+1;
    }
}
```

!!! P/ comparar com algo, ex o texto/símbolo definir (char símbolo);
scanf(" %c", &símbolo); (espaço antes)
e no printf ("%c", símbolo)!!!

Converter p/ maiúsculo:

```
for( =0; texto[i]!='\0'; =i+1){
    if(texto[i]>='a' && texto[i]<='z'){
        texto[i]=texto[i]-32;
    }
}
```

Comparação de textos:

```
scanf("%s\n", senha);
scanf("%s\n", confirma);

for( =0; senha[i]==confirma[i] && senha[i]!='\0' && confirma!='\0';
    =i+1);
if(senha[i]=='\0' && confirma[i]=='\0'){
    printf("Senhas iguais\n");
}
}
```

Concatenação

Juntar variáveis em uma nova variável.

```
1  /* Concatenação */
2
3  #include <stdio.h>
4
5  int main () {
6      char nome[30];
7      char sobrenome[30];
8      char nomeCompleto[60];
9      int
10     printf("Informe seu nome: ");
11     scanf("%s\n", nome);
12
13     printf("Informe seu sobrenome: ");
14     scanf("%s\n", sobrenome);
15                 ↳ espaço
16     for( =0; nome[i]!='\0'; i+=1){
17         nomeComp[i]=nome[i];
18     }
19     for(j=0; sobrenome[j]!='\0'; +=1; +=1){
20         nomeComp[i]=sobrenome[j];
21     }
22
23     nomeComp[i]='\0';
24
25     printf("\nNome completo: %s\n",
26         nomeComp);
27
28     return 0;
29 }
```

Substituindo 19-23 por:

```
19 for( =0; sobrenome[j]!='\0'; +=1) {
20     nomeComp[j+i]=sobrenome[j];
21 }
22
23 nomeComp[j+i]='\0';
```

i já não vai ser igual '\0' !!!

strcat é a função de <string.h>

x = Num++

primeiro

segundo

Ordem de atribuição!

Arrays

<tipo> <identificador> [<tamanho>]

unsigned int vendaPorMes[12]

Inicialização arrays?

int contIdade[5] = {0,0,0,0,0};

~~char~~

unsigned int idade[5];

/* Ler a primeira idade */

scanf("%u", &idade[0]);

Portanto, o array guarda variáveis

```c
/* Ler 4 notas em uma variável e calcular
média */
#include <stdio.h>

int main (){
    double nota[4];
    double media;
    int ;

    for ( = ; < ; ++) {
        printf(" ", +);
        scanf(" ", &nota[i]);
    }
    for ( = ; media = media + nota[i]; < ; ++){
    }
    media = soma / 4.0;

    printf(" MEDIA: %.2f\n", media);
    printf(" notas abaixo da media:\n");

    for ( = ; < ; ++){
        printf(" Nota %d: %.2f\n", +,
        nota[i]);

    return 0;
}
```

* Pode usar #define p/ a entrada ter menos valores, ou mais. Entrada fixa!!!

```c
/* Ler na ordem inversa */

#include <stdio.h>
#define QTDEPRECOS
int main () {
    double preco[QTDEPRECOS];
    int ;

    for ( = 0; i < QTDEPRECOS; ++) {
        printf("Informe o preco %d:", +);
        scanf("%lf", &preco[i]);
    }

    for ( = QTDEPRECOS-1; >= 0; --){
        printf("Preco %d: %.2f\n", +,
        preco[i]);
    }

    return 0;
}
```

* Ver exemplo com break nos slides

* Ver exemplo dos dados no PACA!
* Usar do while p/ quando não se tem uma entrada certa
* Em um for dentro de outro for primeiro se resolve o for de dentro completo, p/ voltar no primeiro for

```c
/* Desenhar a matriz de símbolos */

#include <stdio.h>

int main (){
    int tam, i, j;

    printf("Informe o tamanho: ");
    scanf("%d", &tam);

    for (i = 0; i < tam; ++) {
        for (j = 0; j < tam; ++){
            printf("# ");
        }
        printf("\n");
    }
    return 0;
}
```

Matrizes

<tipo> <identificador> [<tam>] [<tam>]

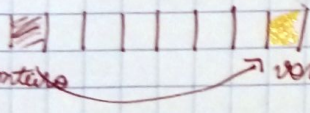
* scanf("%u", &estoque[0][0][0]);

```c
/* Leia e exibir matriz */

#include <stdio.h>
#define TAM_MAX 10

int main(){
    int matriz[TAM_MAX][TAM_MAX];
    int linhas, colunas, i, j;

    printf("                    ");
    scanf("%d", &                );

    printf("                    ");
    scanf("%d", &                );

    for (i = 0; i < linhas; ++i) {
        for (j = 0; j < colunas; ++j) {
            printf("                    ", , );
            scanf("%d", &matriz[i][j]);
        }
    }

    for (i = 0; i < linhas; ++i) {
        for (j = 0; j < colunas; ++j)
            printf("%d ", matriz[i][j]);
        printf("\n");
    }
    return 0;
}
```

Ponteiros

Ponteiros guardam um endereço de memória.

- Código compacto
- Complicações
- Difícil de entender.



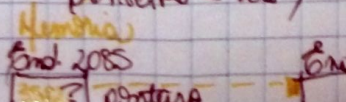
* não necessariamente um byte.

- &: Retorna o endereço de um objeto
- *: Na decl.: informa que é um ponteiro
  Na código, retorna a variável apontada

Código fonte

```c
int *ponteiro;
int idade;
ponteiro = &idade;
*ponteiro = 21;
```

Memória



- printf(ponteiro) imprime o endereço guardado, no caso do exemplo, "3582"!!
- formatador: %p
- Geralmente é do mesmo tipo da variável com o endereço a ser guardado

- O & comercial aponta p/ o endereço da variável.
- (*p2)++; aumenta o endereço
  p2++; aumenta o valor
- O tamanho dos ponteiros é de 4 bits

Campo magnético:
Os nachos umantizados apontam p/ um auto local com seus spins determinados de acordo com o ângulo definido como a inclinação do campo no época.
spins: ponteiro
inclinação: valor da variável
campo: endereço da variável
nacho: endereço do ponteiro.

```c
#include <stdio.h>

int main (){
    double x;
    double *p;

    printf("Informe o valor (inc:m ");
    scanf("%lf", &x);

    p = &x;

    *p = *p * 254;

    printf("%p\n", *p);

    return 0;
}
```

Arquivo

- FILE é o tipo de dados utilizado
- função para abrir um arquivo

- name: é o caminho pro arquivo
- mode:
  - r → ler
  - w → escrever
  - a → add informações
  - r+ → pode ler e atualizar
  - w+ → escrever e atualizar
  - a+ → pode atualizar no final

* fopen → *fclose

06.02.2018

```c
/*Ler msgs e gravar orgns. em TXT*/

#include <stdio.h>
#define TRUE 1
int main (){
    char msg[1000];
    FILE *arquivo;

    arquivo=fopen("msg.txt","w");

    while(TRUE){
        fgets(msg,1000,stdin); -> lê a mensagem
        if(msg[0]=='\n'){
            break;
        }
        fprintf(arquivo,"%s",msg); -> grava a msg com o org.
    }
    fclose(arquivo);

    return 0;
}
```

```c
/*Ler msgs de orgs. TXT e contam de las do linha*/

#include<stdio.h>
#define TRUE 1

int main(){
    char msg[1000];
    FILE *arquivo;

    arquivo=fopen("msg.txt","r");

    while(fgets(msg,1000,arquivo)!=NULL){
        printf("Texto desta linha: %s\n",msg);
    }
    fclose(arquivo);

    return 0;
}
```

```c
/*Ler nome e salário. Gravar em TXT*/

#include <stdio.h>

int main(){
    char nome[100];
    double salario;
    int idade;
    FILE *arquivo;

    printf("Informe seu nome:\n");
    scanf("%[^\n]",&nome[0]);
    printf("Informe sua idade:\n");
    scanf("%d",&idade);
    printf("Informe seu salário:\n");
    scanf("%lf",&salario);

    arquivo=fopen("dados.txt","a");

    fprintf(arquivo,"%s\n",nome);
    fprintf(arquivo,"%d %.2f",idade,salario);

    fclose(arquivo);

    return 0;
}
```

Desenhar Tabuleiro

02.02.2016

```
6 8
1 1 1 1 1 1 1 1
1 0 0 0 0 0 0 1
1 0 1 1 0 1 0 1
1 0 0 0 0 1 0 1
1 0 1 1 0 0 0 1
1 1 1 1 1 1 1 1
```

```c
#include <stdio.h>

#define MAX 30

int main(){
    int tabuleiro[MAX][MAX];
    int linhas,colunas,i,j;
    FILE *arq;

    arq=fopen("fase1.txt","r");

    fscanf(arq,"%d",&linhas); -> eu quero ler
    fscanf(arq,"%d",&colunas);

    for(i=0;i<linhas;++i){
        for(j=0;j<colunas;++j){
            fscanf(arq,"%d",&tabuleiro[i][j]);
        }
    }
    fclose(arq);
```

```c
    printf("\n Tabuleiro lido: \n");

    for(i=0; i<linhas; ++){
        for(j=0; j<colunas; ++){
            printf("%3d\n", tabuleiro[i][j]);
        }
    }
    return 0;
}

/* Imprimir os desenhos do tabuleiro */
for(i=0; i<linhas; ++){
    for(j=0; j<colunas; ++){
        if(tabuleiro[i][j]==0)
        {
            DrawBitmap("grama.bmp", *, *);
        } else {
            DrawBitmap("pareded.bmp", *, *);
        }
    }
}
```

Aleatoriedade

Método dos congruentes lineares (simples)

$$X_{i+1} = (a * X_i + b) \bmod m$$

- a, b e m são var. ints. previamente escolhidas
- X_0 semente
- X_i e X_{i+1} valores entre 0 e m-1.

```c
/* Números aleatórios */

#include <stdio.h>

int main(){
    unsigned int a, b, m, semente, num, i;

    a=;
    b=;
    m=15;
    semente=;
    num=semente;
    for(i=0; i<15; ++){
        num=(a*semente+b)%m;
        printf("%u\n", num);
    }
    return 0;
}
```

```c
#include <time.h>

time_t semente;

time(&semente);
// usar aleatoriedade na semente.
```

- .alos -> função para extrair módulo.
- void não retorna nada
- return responde/encerra o bloco.
- return sem autoreferência (já é "ponteiro").

Funções

- Subrotina
- Encapsular processos
- Esconder detalhes
- Dividir em pequenos fragmentos
- Escrever uma única vez
- É necessário ter sua missão bem definida.
- Não necessariamente função matemática.

```c
/* Criar função p/ exibir menu na tela */

#include <stdio.h>

int exibirMenu(){   // Tipo de retorno; Identificador
void exibirMenu(){
    printf("\n Menu \n");
    printf("1 - Converter pol p/ cm\n");
    printf("2 - Converter cm p/ pol\n");
    printf("3 - Converter Km p/ m\n");
    printf("4 - Converter m p/ Km\n");
    printf("5 - Sair\n");
    printf("\n Escolha uma opção: \n");
}

int main(){
    int opcao;
    exibirMenu(){

    do {
        exibirMenu();
        scanf("%d", &opcao);
    } while(opcao != 5);

    return 0;
}

/* Exemplo de função com parâmetro
double polParaCm(double polegadas){
    double cm;
    cm = polegadas * 2.54;
```

```
    return 0;
}
```

19.09.2018

```
/* Criar função que converte letra p/ Maius. */

#include <stdio.h>

char convLetraMaiusc(char letra){
    char letraMaiusc;

    if(letra>='a' && letra<='z'){
        letraMaiusc= letra-32;
    } else {
        letraMaiusc=letra;
    }
    return letraMaiusc;
}
int main (){
    char simb;
    char simbMaiusc;

    scanf("%c", &simb);

    simbMaiusc = convLetraMaiusc(simb);

    printf("Letra maiuscula: %c", simbMaiusc);

    return 0;
}
```

```
#include <stdio.h>

void troca(int *p1, int *p2){
    int aux;

    aux = *p1;
    *p1 = *p2;
    *p2 = aux;
}
int main(){
    int num1, num2;

    troca(&num1, &num2);

    return 0;
}
```

Trocar valor com valor

Cabeçalhos

10.09.2018

→ Criar cabeçalhos p/ não importar a ordem dos blocos.

- .h → de cabeçalho em inglês.

Como criar bibliotecas?

- Crie um arquivo de header, c/ protótipos de funções.
- Salve como minhaLib.h

```
#ifndef _MINHALIB_H_
#define _MINHALIB_H_

char convLMaiusc (char letra);
void convLMaiusc2 (char *letra);
void convTMaiusc (char txt[100]);

#endif
```

→ Exemplo:

```
#include "bibtextos.h"
```

bibliotecas criadas pelo usuário

```
#include <bibtextos.h>
```

bibliotecas do sistema

→ O mesmo o código-fonte.

stdio.h	ctype.h	math.h
• remove	• toupper(c)	• pow(x,y)
• rename	• tolower(c)	• sqrt(x)
• fread	string.h	• sin(x)
• fwrite	• strcpy	• cos(x)
stdlib.h		• tan(x)
• int abs		• log10(x)
• int rand		• floor(x)

Alguns exemplos!!

Cadastro de livros (Structs)

Coleção de variáveis

```
struct Livro {
    char titulo[100];
    char autor[150];
    int ano;
};
```