1ª lista de exercícios - ICC 1

Prof. Mario Gazziro

Site da disciplina

- http://wiki.icmc.usp.br/index.php/Scc-120%28mat%29

1. O que é um algoritmo?

- Um conjunto de instruções, cujo objetivo é resolver um problema. Este deve conter instruções, uma ordem e um fim.

Os conjuntos de instruções abaixo são algoritmos?

Rotina para ir trabalhar:

1. Acordar
2. Tomar banho
3. Vestir-se
4. Tomar café da manhã
5. Tirar o carro da garagem
6. Ir para o trabalho

Coisas a fazer hoje:

1. Ir à aula
2. Me divertir
3. Fazer esportes
4. Dormir
5. Estudar para a prova

Os conjuntos de instruções abaixo são algoritmos?

Rotina para ir trabalhar:

1. Acordar
2. Tomar banho
3. Vestir-se
4. Tomar café da manhã
5. Tirar o carro da garagem
6. Ir para o trabalho

Coisas a fazer hoje:

- Ir à aula
- Fazer esportes
- Dormir
- Ir a uma festa
- Estudar para a prova

2. Escreva um algoritmo para as seguintes situações:

- Fritar um ovo
- Trocar uma lâmpada
- Sacar dinheiro no banco 24h
- Marcar um consulta no médico

2. Escreva um algoritmo para as seguintes situações:

- Fritar um ovo
- Trocar uma lâmpada
- Sacar dinheiro no banco 24h
- Marcar um consulta no médico

2. Escreva um algoritmo para as seguintes situações:

- Fritar um ovo
- Trocar uma lâmpada
- Sacar dinheiro no banco 24h
- Marcar um consulta no médico

2. Escreva um algoritmo para as seguintes situações:

- Fritar um ovo
- Trocar uma lâmpada
- Sacar dinheiro no banco 24h
- Marcar um consulta no médico

3. Dê exemplos de dispositivos de entrada e saída do computador.

4. O que é um bit? E um byte? Comente brevemente a importância dos números binários na computação.

- Bit: menor unidade de informação, que pode assumir um entre dois valores. Exemplo: verdadeiro e falso, ligado e desligado, um e zero, etc.
- Byte: conjunto de 8 bits.
- A maior importância do sistema binário para a computação reside no fato de que a aritmética/lógica booleana é muito mais simples de implementar em circuitos eletrônicos que qualquer outra.

5. Qual a diferença entre linguagem de máquina, linguagens de baixo nível e linguagens de alto nível? Dê exemplos de linguagens de alto nível.

Uma linguagem de máquina

- Uma linguagem de máquina é um conjunto de instruções que um processador pode executar, as quais são representadas por bits.

Opcode	Instruction	RTN
0000	JnS *X*	MBR ← PC MAR ← *X* M[MAR] ← MBR MBR ← *X* AC ← 1 AC ← AC + MBR PC ← AC
0001	Load *X*	MAR ← *X* MBR ← M[MAR], AC ← MBR
0010	Store *X*	MAR ← *X*, MBR ← AC M[MAR] ← MBR
0011	Add *X*	MAR ← *X* MBR ← M[MAR] AC ← AC + MBR
0100	Subt *X*	MAR ← *X* MBR ← M[MAR] AC ← AC - MBR
0101	Input	AC ← InREG
0110	Output	OutREG ← AC
0111	Halt	
1000	Skipcond	If IR[11–10] = 00 then If AC < 0 then PC ← PC + 1 Else If IR[11–10] = 01 then If AC = 0 then PC ← PC + 1 Else If IR[11–10] = 10 then If AC > 0 then PC ← PC + 1
1001	Jump *X*	PC ← IR[11–0]
1010	Clear	AC 0
1011	AddI *X*	MAR ← X MBR ← M[MAR] MAR ← MBR MBR ← M[MAR] AC ← AC + MBR
1100	JumpI *X*	MAR ← *X* MBR ← M[MAR] PC ← MBR

Linguagens de baixo nível e linguagens de alto nível

- Alto nível: C, Pascal, Java, Basic, etc.
- Baixo nível: Assembly.

```
begin
  clrscr;
  x := 0;
  writeln('Numeros perfeitos abaixo de');
  Readln(ate);
  repeat
    x := x + 1;
    soma := 0;
    for i := 1 to x - 1 do
    begin
      if x mod i = 0 then
        soma := soma + i;
    end;
    if soma = x then
    begin
      writeln(x);
    end;
  until (x > ate);
  writeln('Pressione qualquer tecla para finalizar…');
  readkey;
end.
```

Trecho de um código em Pascal.

```
;Instruçoes com acumulador:
        STM     #1000h, AR1
        LD      #0, A
        STL     A, *AR1
        LD      #1, A
        STL     A, *AR1
        LD      #65535, A
        STL     A, 10
        STH     A, 10
        STL     A, *AR1
        STH     A, *AR1
```

Trecho de um código em Assembly.

6. Quando há pouco espaço em disco disponível e mais memória primária, é melhor trabalhar com programas compilados ou interpretados? Por quê?

- Programas escritos em linguagens interpretadas dependem de um software chamado interpretador para que possam executar, enquanto que linguagens compiladas não.
- A conversão do código para linguagem de máquina é feita em tempo real nas linguagens interpretadas, economizando assim espaço em disco.

7. Qual a diferença entre erros de compilação e erros de execução?

- Erro de compilação (em C):

```
prinf (“Escreva um numero qualquer: );
```

- Correção:

```
printf (“Escreva um numero qualquer: ”);
```

- Erro de execução (trecho em pseudocódigo):

```
Leia (A);
Leia (B);
C = A/B;
```

Que erro poderia ocorrer no código acima?

8. Qual a importância dos comentários no código?

- Organização do código
- Deixar claro o que está sendo feito num determinado trecho do código
- Devemos pensar que outra pessoa pode precisar ler (e entender) nosso código

9. O que o programa abaixo faz?

```
int main()
{
    int n1;
    int n2;

    printf("Entre com dois numeros  naturais:"
    scanf("%d %d", &n1, &n2);
    if(n1 > n2)
        printf("%d", n1);
    else
        printf("%d", n2);

    return 0;
}
```

10. Faca um programa em C para calcular a média aritmética entre 3 provas: p1, p2 e p3.

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

int main ()
{
    float p1, p2, p3, resultado;

    printf ("Escreva as 3 notas: ");
    scanf ("%f %f %f", &p1, &p2, &p3);

    resultado = (p1 + p2 + p3)/3;

    printf("A media: %f \n\n", resultado);

    system ("PAUSE");
    return 0;
}
```