Trabalho de Linguagem de Programação I.

Universidade Federal de Santa Catarina.

Professora: Olga

Alunos : Matheus Francisco B Machado Nº 14202492

Augusto Lutz Nº 14204416

Classificação de linguagem de programação.

São classificadas como , Linguagem de programação de primeira geração , Linguagem de programação de multiparadigma , Linguagem de programação de terceira geração, Linguagem de programação de segunda geração , Linguagem de programação de quarta geração , Linguagem de programação de baixo nivel e Linguagem de programação de alto nivel.

Exemplos

1. **Linguagem de máquina** – Todos os computadores possuem no seu processador um conjunto de códigos que podem executar expresso em números binários. Um programa em código de máquina consiste numa sequencia de números em sistema binário, hexadecimal ou octal contendo as instruções que o computador deve executar. Hoje em dia ninguém programa em código de máquina, mas esta era a forma de programação usada nos primeiros computadores.

2. **Linguagem de montagem**  – A linguagem de montagem ou assembler está um degrau acima da linguagem de máquina. As instruções brutas do código de máquina são substituidas por simblos mnemônicos destas instruções. Como existe uma relação direta entre a linguagem de máquina e a linguagem de montagem esta não é compilada e a passagem para linguagem de máquina é realizada por um montador conhecido em inglês como assembler. A linguagem de montagem, como a linguagem de máquina é especifica para cada máquina.

Os dois tipos de linguagem de programação acima são conhecidos como linguagens de baixo nível. A linguagem de máquina é uma linguagem de baixo nível de primeira geração e a linguagem de montagem é considerada de segunda geração.

3. **Linguagem de alto nível -**  Neste tipo de linguagem as instruções são dadas em linguagem humana respeitada, é claro, as idiossincrasias da linguagem. Neste caso, o proprio computador se encarrega em passara as instruções para a linguagem de máquina. Este foi um grande passo, pois abriu o uso dos computadores para os cientistas engenheiros e administradores. As linguagens de alto nível podem ser interpretadas ou compiladas. Ela será interpretada se o computador for traduzindo para linguagem de máquina cada instrução a medida em que for executando. Ela será compilada se todas as instruções do programa forem traduzidas para a linguagem de máquina antes da execução do programa. Neste caso o programa em linguagem de alto nível é conhecido como programa fonte e o programa em linguagem de máquina é conhecido como programa objeto. São exemplos o Fortran, o Basic, o C++, etc.

Definição de algoritmo

- Algoritmo é umas receita mostrando o passo a passo , procedimentos necessários para a resolução de uma tarefa, não responde a pergunta “o que fazer ?” mas sim (como fazer). Em termos mais técnicos, definimos com uma sequência lógica , finita e definida de instruções que são seguidas para resolver um problema ou executar uma tarefa.

Muitas vezes executamos algoritmos

muitas vezes inconscientemente executamos algoritmos. Mas o que é Algoritmo? É simplesmente uma “receita” para executarmos uma tarefa ou resolver algum problema. E como toda receita, um algoritmo também deve ser finito. Se seguirmos uma receita de bolo corretamente, conseguiremos fazer o bolo. A computação utiliza muito esse recurso, então se você pretende aprender programação, obviamente deve saber o que é algoritmo.

Problema 01- Dado um número inteiro qualquer e um digito especificado pelo usuário, escreva o programa documentado(\*) que calcula quantas vezes aquele digito aparece naquele número.

// FOR COURSE: Programação em Computadores I.

// DATE: May 12, 2015.

//

// PURPOSE:

// Receba um numero inteiro e um digito especificado pelo usuário

// calcule quantas vezes o digito apareceu no numero.

//

// OVERALL METHOD:

// O resultado é calculado de acordo com a fórmula:

// Digito == X%10;

// X = X/10;

// Esse programa faz controle de dados de entrada comparando enquanto os números de entrada não forem igual a -1.

//

// FUNCTIONS:

//

// INCLUDED FILES: stdio.h

//

// DATA FILES: Não

//

// PARAMETERS:

// nome tipo value/reference descrição

// -------------------------------------------------------

// X int value Numero inteiro

// XE int value Escolher o Digito especifico

// Digito int value Armazenar o digito especifico

// Contador int value Contar a quantidade de vezes digitadas

//

#include <stdio.h>

int main ()

{

int X;

int XE;

int Digito;

int Contador;

Contador = 0;

printf ("\n Digite um numero inteiro: ");

scanf ("%d",&X);

printf("\n Digite o numero especifico para contagem: ");

scanf("%d",&XE);

while (X != 0)

{

Digito == X%10;

X = X/10;

if (Digito == XE){

Contador++;

}

}

printf ("\n O digito %d foi digitado %d vezes", XE, Contador);

return 0;

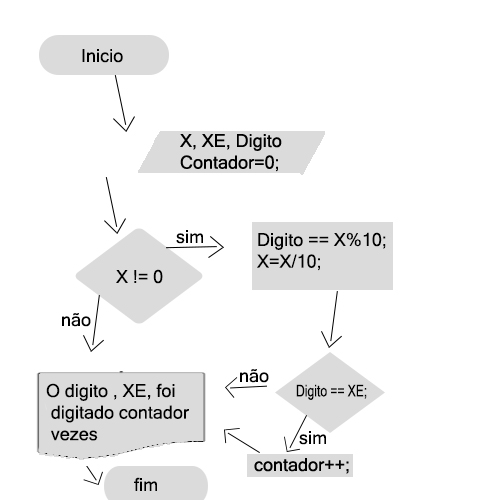
}

// EXAMPLE INPUT AND OUTPUT

// O usuário digita o um numero :13211

// O usuário digita um digito especifico: 1

// A quantidade de numero 1 que tem no numero digitado é: 3



Problema 2: Escreva o programa documentado(\*) que receba códigos dos produtos com as respectivas quantidades. Feita a entrada de dados é calculado o preço total da nota e o valor do desconto. Os preços dos produtos estão especificados em Tabela 1. O valor do desconto é calculado de acordo com a Tabela 2 e é aplicado sobre o preço total da nota;

// FOR COURSE: Programação em Computadores I.

// DATE: May 13, 2015.

//

// PURPOSE:

// Receba códigos dos produtos com as respectivas quantidades até que um código especificado é

// informado. Assim que entrar os dados, calcular o preço total da nota e o valor do desconto.

//

// OVERALL METHOD:

// O resultado é calculado de acordo com a fórmula:

// TotalP = TotalP + (10\*QuantidadeP);

// TotalP = TotalP = (15\*QuantidadeP);

// TotalP = TotalP = (20\*QuantidadeP);

// TotalP = TotalP = (30\*QuantidadeP);

// E o resulado do desconto é calculado de acordo com a fórmula:

// Desconto = TotalP \* 0.05; TotalDsc = TotalP - Desconto;

// Desconto = TotalP \* 0.10; TotalDsc = TotalP - Desconto;

// Desconto = TotalP \* 0.15; TotalDsc = TotalP - Desconto;

// Esse programa faz controle de dados de entrada comparando enquanto os números de entrada não forem igual a -1.

//

// FUNCTIONS:

//

// INCLUDED FILES: stdio.h

//

// DATA FILES: Não

//

// PARAMETERS:

// nome tipo value/reference descrição

// -------------------------------------------------------

// CodigoP int value Codigo de produto comprado

// QuantidadeP int value Quantidade do produto

// TotalP int value Total de produtos

// TotalDsc float value Total do desconto

// Desconto float value Desconto

#include <stdio.h>

int main()

{

int CodigoP; // Codigo de produto comprado

int QuantidadeP; // Quantidade do produto

int TotalP; // Total de produtos

float TotalDsc; // Total do desconto

float Desconto; // Desconto

TotalP = 0;

CodigoP = 0;

while ( CodigoP != - 1 )

{

printf ("\n Digitar o codigo do produto: ");

scanf ("%i", &CodigoP);

printf ("\n Digite a quantidade do produto: ");

scanf ("%i", &QuantidadeP);

if (CodigoP >= 1 && CodigoP <= 10)

TotalP = TotalP + (10\*QuantidadeP);

if (CodigoP >= 11 && CodigoP <= 20)

TotalP = TotalP = (15\*QuantidadeP);

if (CodigoP >= 21 && CodigoP <= 30)

TotalP = TotalP = (20\*QuantidadeP);

if (CodigoP >= 31 && CodigoP <= 40)

TotalP = TotalP = (30\*QuantidadeP);

}

if (TotalP <= 250)

{Desconto = TotalP \* 0.05;

TotalDsc = TotalP - Desconto;}

if (TotalP > 250 && TotalP <= 500)

{Desconto = TotalP \* 0.10;

TotalDsc = TotalP - Desconto;}

if (TotalP > 500)

{Desconto = TotalP \* 0.15;

TotalDsc = TotalP - Desconto;}

printf ("\n Valor Total da Nota sem Desconto; %i", TotalP);

printf ("\n Total de Desconto: %.2f", Desconto);

printf ("\n Valor Total da Nota com Desconto: %.2f", TotalDsc);

return 0;

}

// EXAMPLE INPUT AND OUTPUT

// O usuário digita o codigo do produto: 1

// O usuário digita a quantidade do produto: 10

// O usuário digita o codigo do produto: -1

// Valor total da nota sem desconto: 100

// Total de desconto: 5.00

// Valor total da nota com desconto: 95.00

