(/notific

Algoritmos e Programação Estruturada (/alu...

Av1 - Algoritmos e Programação Estruturada

Sua avaliação foi confirmada com sucesso

Informações Adicionais

Período: 31/07/2023 00:00 à 13/11/2023 23:59
Situação: Cadastrado
Tentativas: 1 / 3
Pontuação: 1000
Protocolo: 921110465

Avaliar Material

1) A lógica de programação pode ser definida como uma técnica de encadear pensamentos para atingir determinado objetivo. Ela faz-se necessária para desenvolver programas e sistemas, pois permite definir a sequência lógica para a solução de um problema.

```
1  var
     real: x, y, z;
2  Início
     escreva (“Digite um Numero:”);
     leia x;
3
     escreva (“Digite outro Numero:”);
     leia y;
     z ¿ x + z + 1;
4
     escreva(“Resultado = ”, z);
5  Fim.
```

Considerando o algoritmo apresentado, julgue as afirmações que seguem e marque (V) para verdadeiro ou (F) para falso.

( ) A parte 1 do pseudocódigo indica a declaração das variáveis que são compatíveis com qualquer tipo de dados.

( ) A parte 3 do pseudocódigo corresponde a entrada de dados do algoritmo.

( ) A parte 4 do pseudocódigo corresponde tando a parte de processamento quanto a parte de saída de dados do algoritmo..

Agora, assinale a alternativa que contém sequência correta.

Alternativas:

a) F - V - V. Alternativa assinalada

b) F - V - F.

c) V - V - V.

d) V - V - F.

e) F - F - F.

2) Variáveis com estruturas compostas do tipo vetor ou matriz, só são capazes de armazenar valores de um mesmo tipo, porem, além das estruturas homogêneas as linguagens de programação oferecem um tipo de variável composta heterogênea chamada de estruturas (structs) ou ainda de registros.

Na linguagem C, a criação de uma estrutura deve ser feita antes da função main() e deve possuir a seguinte sintaxe:

Alternativas:

a) struct <nome>; <tipo> <nome_da_variavel1>; <tipo> <nome_da_variavel2>; ... ;

b) struct { <tipo> <nome_da_variavel1>; <tipo> <nome_da_variavel2>; ... };

c) struct <nome>{ <nome_da_variavel1>; <nome_da_variavel2>; ... };

d) struct <nome>{ <tipo> <nome_da_variavel1>; <tipo> <nome_da_variavel2>; ... }; Alternativa assinalada

e) <nome> struct{ <tipo> <nome_da_variavel1>; <tipo> <nome_da_variavel2>; ... };

3) A criação de um ponteiro só faz sentido se for associado a algum endereço de memória, para isso usa-se a seguinte sintaxe:
1. int idade = 18;
2. int *ponteiro_para_idade = &idade;

Na linha 1 criamos uma variável primitiva inteira com valor 18 e na linha 2 associamos um ponteiro chamado ponteiro_para_idade ao endereço da variável primitiva idade.

Podemos imprimir o conteúdo do ponteiro, que será o endereço da variável que ele aponta. Utilizando o ponteiro criado anteriormente (ponteiro_para_idade) temos a seguinte sintaxe:

Alternativas:

a) printf("\n Conteudo do ponteiro: %d", &idade);

b) scanf("\n Conteudo do ponteiro: %p", ponteiro_para_idade);

c) gets("\n Conteudo do ponteiro: %d", &idade);

d) printf("\n Conteudo do ponteiro: %p", ponteiro_para_idade); Alternativa assinalada

e) scanf("\n Conteudo do ponteiro: %d", &idade);

4) O laço do-while executa, pelo menos uma vez, o que está dentro dele e só ao final da execução é que ele faz o teste, usando o velho e conhecido laço while. Ou seja, tem-se a garantia que o laço vai ser executado uma vez, sem precisar inicializar variável ou pedir dados ao usuário antes do while. Vale lembra que do, em inglês e nesse contexto, do significa "faça" e while significa "enquanto".

- Fonte:Disponível em<Adaptado de https://www.cprogressivo.net/2013/02/O-que-e-e-como-usar-o-laco-DO-WHILE-em-linguagem-C.html>Acesso.16.Jul.2018.

Esse laço do-while quer dizer:

Alternativas:

a) "faça isso" -> código -> "enquanto essa condição for verdadeira, repita". Alternativa assinalada

b) "faça aquilo" -> código -> "enquanto essa condição for verdadeira, repita".

c) "faça isso" -> laço -> "enquanto essa condição for falsa, repita".

d) "faça aquilo" -> laço -> "se essa condição for verdadeira, pare".

e) "faça isso" -> código -> "se essa condição for verdadeira, pare".

5) Cada programa C tem uma função principal que deve ser nomeada main. Se o seu código obedece ao modelo de programação Unicode, você pode usar a versão de main para caracteres largos, wmain. A função main serve como o ponto de partida para a execução do programa. Em geral, ela controla a execução direcionando as chamadas para outras funções no programa. Normalmente, um programa para de ser executado no final de main, embora possa ser terminado em outros pontos por diversos motivos.

Para o programa apresentado a seguir retornar apenas os números pares de 1 a 10 deverá possuir alguns parâmetros, complete as lacunas da sentença a seguir.

```
#include <stdio.h>

int main( )
{
int count = _____;

while(count <= _____)
    {
if(count%_____ == 0)
printf("%d ",count);

count++;
    }
}
```

Assinale a alternativa que completa as lacunas corretamente.

Alternativas:

a) 1 / 10 / 2. Alternativa assinalada

b) 0 / 9 / 1.

c) 1 / 9 / 2.

d) 0 / 10 / 2.

e) 0 / 10 / 1.