Trabalho de linguagem de programação

1º semestre

1 -

DEFINIÇÃO DO PROBLEMA

O primeiro exercício pede para que seja imprimido as iniciais do nome do desenvolvedor do programa utilizando o caractere “#” em uma escala de 6x6.

SOLUÇÃO

Para a impressão correta do exercício o código foi feito imprimindo a primeira fileira de todas as iniciais utilizando um printf(“”,) com um \n no final, para a quebra de linha, e prosseguir com a segunda linha utilizando o mesmo método da primeira, até a sexta linha.

VALIDAÇÃO

Para ter confirmação da correta impressão foi analisado se houve algum caractere mal posicionado após a execução do programa.

COMENTARIO E CONCLUSÕES

O código ficou um pouco grande, mas cumpriu seu propósito.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main(void)

{

char a;

a= '#';

printf("%c%c%c%c%c %c%c%c%c%c%c %c %c\n%c %c %c%c %c %c\n%c %c %c%c %c %c\n%c %c %c%c %c %c\n%c %c %c%c %c %c\n%c%c%c%c%c %c%c %c%c%c%c%c%c\n\n", a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a, a);

system("pause");

return 0;

}

Resultados:

##### ###### # #

# # ## # #

# # ## # #

# # ## # #

# # ## # #

##### ## ######

Pressione qualquer tecla para continuar. . .

2 -

DEFINIÇÃO DO PROBLEMA

O primeiro código do segundo exercício (INTASIGN.C) pedia para que seja impresso cada cálculo feito dentro do código

O segundo código do segundo exercício (LOTTYPES.C) entrega 7 variáveis e mostra como deveria ser impressa e pedia que fosse corrigida o número de casas decimais e a forma na qual foi impressa em algumas linhas.

O terceiro código do segundo exercício (MORTYPES.C) entregava variáveis, contas que mudavam o valor das variáveis e pedia que fosse impresso as variáveis corretamente.

SOLUÇÃO

Para a resolução do INTASIGN.C foi adicionado o #include<stdio.h>, #include<stdlib.h> e o system(“pause”) e a impressão da primeira letra de cada cálculo, até o 9º cálculo e a impressão de qualquer variável das últimos três contas pois iriam dar o mesmo resultado independente da variável impressa.

Para a resolução do LOTTYPES.C foi adicionado o #include<stdlib.h> e o system(“pause”) e corrigido 4 linhas do código sendo eles: printf("c = %d\n", c); que foi trocado por printf("c = %hd\n", c);, printf("g = %f\n", g); que foi trocado por printf("g = %lf\n", g);, printf("a = %-7d\n", a); que foi trocado por printf("a = %-7.3d\n", a); e printf("f = %12.5f\n", f); que foi trocado por printf("f = %11.5f\n", f);.

Para a resolução do MORTYPES.C foi adicionado o #include<stdio.h>, #include<stdlib.h> e o system(“pause”) e feito a adição do printf(“”,), com atenção na forma que seriam impressas, sendo elas %d, %d, %.2f e %d.

VALIDAÇÃO

Para ter a confirmação dos testes foi analisado as saídas de cada programa após a correção e a comparação das saídas com o que foi pedido no exercício.

COMENTÁRIOS E CONCLUSÕES

Nos programas INTASIGN.C e MORTYPES.C não houve erros durante a correção, mas no LOTTYPES.C foi entendido que o exercício pedia que a saída ficasse igual o resultado dado no final do código.

LISTAGENS

Código comentado: INTASIGN.C

#include<stdio.h>

#include<stdlib.h>

/\* Chapter 4 - Program 1 - INTASIGN.C \*/

/\* This program will illustrate the assignment statements \*/

int main()

{

int a, b, c; /\* Integer variables for examples \*/

a = 12;

b = 3;

c = a + b; /\* simple addition \*/

printf("%d\n",c);

c = a - b; /\* simple subtraction \*/

printf("%d\n",c);

c = a \* b; /\* simple multiplication \*/

printf("%d\n",c);

c = a / b; /\* simple division \*/

printf("%d\n",c);

c = a % b; /\* simple modulo (remainder) \*/

printf("%d\n",c);

c = (12\*a + b/2 - a\*b\*2)/(a\*c + b\*2);

printf("%d\n",c);

c = (c/4+13\*(a + b))/(3 - a\*b + 2\*a\*a);

printf("%d\n",c);

a = a + 1; /\* incrementing a variable \*/

printf("%d\n",a);

b = b \* 5;

printf("%d\n",b);

a = b = c = 20; /\* multiple assignment \*/

printf("%d\n",a);

a = b = c = (a + b \* c/ 3);

printf("%d\n",a);

a = (b = (c = 20)); /\* Identical to line 22 \*/

printf("%d\n",a);

system ("pause");

return 0;

}

/\* Result of execution

(No output from this program.)

\*/

Resultado:

15

9

36

4

0

12

0

13

15

20

153

20

Pressione qualquer tecla para continuar. . .

Código comentado: LOTTYPES.C

/\* Chapter 4 - Program 3 - LOTTYPES.C \*/

#include <stdio.h>

#include<stdlib.h>

int main()

{

int a; /\* simple integer type \*/

long int b; /\* long integer type \*/

short int c; /\* short integer type \*/

unsigned int d; /\* unsigned integer type \*/

char e; /\* character type \*/

float f; /\* floating point type \*/

double g; /\* double precision floating point \*/

a = 1023;

b = 2222;

c = 123;

d = 1234;

e = 'X';

f = 3.14159;

g = 3.1415926535898;

printf("a = %d\n", a); /\* decimal output \*/

printf("a = %o\n", a); /\* octal output \*/

printf("a = %x\n", a); /\* hexadecimal output \*/

printf("b = %ld\n", b); /\* decimal long output \*/

printf("c = %hd\n", c); /\* decimal short output \*/

printf("d = %u\n", d); /\* unsigned output \*/

printf("e = %c\n", e); /\* character output \*/

printf("f = %f\n", f); /\* floating output \*/

printf("g = %lf\n", g); /\* double float output \*/

printf("\n");

printf("a = %d\n", a); /\* simple int output \*/

printf("a = %7d\n", a); /\* use a field width of 7 \*/

printf("a = %-7.3d\n", a); /\* left justify in field of 7 \*/

c = 5;

d = 8;

printf("a = %\*d\n", c, a); /\* use a field width of 5 \*/

printf("a = %\*d\n", d, a); /\* use a field width of 8 \*/

printf("\n");

printf("f = %f\n", f); /\* simple float output \*/

printf("f = %12f\n", f); /\* use field width of 12 \*/

printf("f = %12.3f\n", f); /\* use 3 decimal places \*/

printf("f = %11.5f\n", f); /\* use 5 decimal places \*/

printf("f = %-12.5f\n", f); /\* left justify in field \*/

system("pause");

return 0;

}

/\* Result of execution

a = 1023

a = 1777

a = 3ff

b = 2222

c = 123

d = 1234

e = X

f = 3.141590

g = 3.141593

a = 1023

a = 1023

a = 1023

a = 1023

a = 1023

f = 3.141590

f = 3.141590

f = 3.142

f = 3.14159

f = 3.14159

\*/

Resultado:

a = 1023

a = 1777

a = 3ff

b = 2222

c = 123

d = 1234

e = X

f = 3.141590

g = 3.141593

a = 1023

a = 1023

a = 1023

a = 1023

a = 1023

f = 3.141590

f = 3.141590

f = 3.142

f = 3.14159

f = 3.14159

Pressione qualquer tecla para continuar. . .

Código comentado: MORTYPES.C

/\* Chapter 4 - Program 2 - MORTYPES.C \*/

/\* The purpose of this file is to introduce additional data types \*/

#include<stdio.h>

#include <stdlib.h>

int main()

{

int a, b, c; /\* -32768 to 32767 (16bits, ou short int) no decimal point \*/

/\* -2 147 483 648 to 2 147 483 647 (int de 32bits) ou long int \*/

char x, y, z; /\* -128 to 127 with no decimal point \*/

float num, toy, thing; /\* 3.4E-38 to 3.4E+38 with decimal point \*/

a = b = c = -27;

x = y = z = 'A';

num = toy = thing = 3.6792;

a = y; /\* a is now 65 (character A) \*/

printf("%d\n", a);

x = b; /\* x is now -27 \*/

printf("%d\n" , x);

num = b; /\* num will now be -27.00 \*/

printf("%.2f\n", num);

a = toy; /\* a will now be 3 \*/

printf("%d\n\n", a);

system("pause");

return 0;

}

/\* Result of execution

(No output from this program.)

\*/

Resultado:

65

-27

-27.00

3

Pressione qualquer tecla para continuar. . .

3 -

DEFINIÇÃO DO PROBLEMA

O terceiro exercício entregava três variáveis, pedia para que fossem impressas utilizando %d, %x, %u, %c, %f e %lf, fosse apontado os possíveis erros ao imprimir as variáveis e analisar as saídas.

SOLUÇÃO

A solução do exercício foi adicionar vários printf(“”) com suas % corretas

VALIDAÇÃO

Para ter a confirmação dos testes foi analisado a saída de cada printf(“”) e sua %

COMENTÁRIOS E CONCLUSÕES

Na impressão do código desse exercício foi mostrado que o %d imprime o valor em decimal inteiro, %x imprime em hexadecimal inteiro, %u em usigned, %c em caractere, “%f” em float e %lf sendo double ou long float. Na execução do programa o printf(“%f %f”, a, d) e printf(“%lf %lf”, a, d) foram zerados.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main()

{

// varaiveis

int a=-123;

char c='A';

double d=3.1415926535898;

printf("Em decimal.\n%d\n%d\n%d\n\n", a, c, d);

printf("Em hexadecimal.\n%x\n%x\n%x\n\n", a, c, d);

printf("Em unsigned.\n%u\n%u\n%u\n\n", a, c, d);

printf("Em character.\n%c\n%c\n%c\n\n", a, c, d);

printf("Em float.\n%f\n%f\n%f\n\n", a, c, d);

printf("Em double float.\n%lf\n%lf\n%lf\n\n", a, c, d);

system("pause");

return 0;

}

Resultado:

Em decimal.

-123

65

1413754152

Em hexadecimal.

ffffff85

41

54442d28

Em unsigned.

4294967173

65

1413754152

Em character.

à

A

(

Em float.

0.000000

3.141593

0.000000

Em double float.

0.000000

3.141593

0.000000

Pressione qualquer tecla para continuar. . .

4 -

DEFINIÇÃO DO PROBLEMA

O problema se define em corrigir expressões e imprimir os resultados delas.

SOLUÇÃO

Em algumas das expressões do problema para chegar ao resultado é necessário acrescentar parênteses para obter a precedência correta e acrescentar printf() a cada expressão para imprimi-las.

VALIDAÇÃO

Para ter a confirmação dos testes foi adicionado printf(), assim temos os resultados das expressões e podemos comparar com as contas feitas manualmente.

x=(2+3)\*6=30

y=5/(2\*3) =0

i=j=(3+2)/4=1

a=3+2\*(b=7/2)=9

c=5+10%2/3=5

d=-(3%5)\*3/2= -4

x= -3+4\*5-6= 11

x=(7+6)%5/2=1

COMENTÁRIOS E CONCLUSÕES

De acordo com a teoria, o resultado foi o esperado.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main()

{

// variaveis

int x, y, i, j, a, b, c, d, h, s;

// contas

j= ((3+2)/4);

b= (7/2);

x= (2+3)\*6;

y= 5/(2\*3);

i= j;

a= 3+2\*b;

c= 5+10%2/3;

d= -(3%5)\*3/2;

h= -3+4\*5-6;

s= (7+6)%5/2;

// impressao dos resultados

printf("i x=%d\n", x);

printf("ii y=%d\n", y);

printf("iii i=%d\n", i);

printf("iv a=%d\n", a);

printf("v c=%d\n", c);

printf("vi d=%d\n", d);

printf("vii x=%d\n", h);

printf("viii x=%d\n", s);

system("pause");

return 0;

}

Resultados:

i x=30

ii y=0

iii i=1

iv a=9

v c=5

vi d=-4

vii x=11

viii x=1

5 -

DEFINIÇÃO DO PROBLEMA

O exercício pede que façamos um programa que imprima os números e os caracteres da tabela ASCII

SOLUÇÃO

Utilizamos o comando de repetição, for, e um printf() para imprimir os resultados

VALIDAÇÃO

Para verificar o resultado cada caractere foi comparado com seu respectivo número.

COMENTÁRIOS E CONCLUSÕES

De acordo com a teoria os resultados obtidos na impressão foram atendidos

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main(void)

{

int carac, num; //variáveis

for(carac=0 ; carac<=127 ; carac++) //comando para somar +1 em carac até carac=127

{

printf(" %d corresponde ao caractere %c\n", carac,carac); //comando imprimir número e o caractere

}

return 0;

}

Resultados:

0 corresponde ao caractere

1 corresponde ao caractere ☺

2 corresponde ao caractere ☻

3 corresponde ao caractere ♥

4 corresponde ao caractere ♦

5 corresponde ao caractere ♣

6 corresponde ao caractere ♠

7 corresponde ao caractere

8 corresponde ao caractere

9 corresponde ao caractere

10 corresponde ao caractere

11 corresponde ao caractere ♂

12 corresponde ao caractere ♀

13 corresponde ao caractere

14 corresponde ao caractere ♫

15 corresponde ao caractere ☼

16 corresponde ao caractere ►

17 corresponde ao caractere ◄

18 corresponde ao caractere ↕

19 corresponde ao caractere ‼

20 corresponde ao caractere ¶

21 corresponde ao caractere §

22 corresponde ao caractere ▬

23 corresponde ao caractere ↨

24 corresponde ao caractere ↑

25 corresponde ao caractere ↓

26 corresponde ao caractere →

27 corresponde ao caractere ←

28 corresponde ao caractere ∟

29 corresponde ao caractere ↔

30 corresponde ao caractere ▲

31 corresponde ao caractere ▼

32 corresponde ao caractere

33 corresponde ao caractere !

34 corresponde ao caractere "

35 corresponde ao caractere #

36 corresponde ao caractere $

37 corresponde ao caractere %

38 corresponde ao caractere &

39 corresponde ao caractere '

40 corresponde ao caractere (

41 corresponde ao caractere )

42 corresponde ao caractere \*

43 corresponde ao caractere +

44 corresponde ao caractere ,

45 corresponde ao caractere -

46 corresponde ao caractere .

47 corresponde ao caractere /

48 corresponde ao caractere 0

49 corresponde ao caractere 1

50 corresponde ao caractere 2

51 corresponde ao caractere 3

52 corresponde ao caractere 4

53 corresponde ao caractere 5

54 corresponde ao caractere 6

55 corresponde ao caractere 7

56 corresponde ao caractere 8

57 corresponde ao caractere 9

58 corresponde ao caractere :

59 corresponde ao caractere ;

60 corresponde ao caractere <

61 corresponde ao caractere =

62 corresponde ao caractere >

63 corresponde ao caractere ?

64 corresponde ao caractere @

65 corresponde ao caractere A

66 corresponde ao caractere B

67 corresponde ao caractere C

68 corresponde ao caractere D

69 corresponde ao caractere E

70 corresponde ao caractere F

71 corresponde ao caractere G

72 corresponde ao caractere H

73 corresponde ao caractere I

74 corresponde ao caractere J

75 corresponde ao caractere K

76 corresponde ao caractere L

77 corresponde ao caractere M

78 corresponde ao caractere N

79 corresponde ao caractere O

80 corresponde ao caractere P

81 corresponde ao caractere Q

82 corresponde ao caractere R

83 corresponde ao caractere S

84 corresponde ao caractere T

85 corresponde ao caractere U

86 corresponde ao caractere V

87 corresponde ao caractere W

88 corresponde ao caractere X

89 corresponde ao caractere Y

90 corresponde ao caractere Z

91 corresponde ao caractere [

92 corresponde ao caractere \

93 corresponde ao caractere ]

94 corresponde ao caractere ^

95 corresponde ao caractere \_

96 corresponde ao caractere `

97 corresponde ao caractere a

98 corresponde ao caractere b

99 corresponde ao caractere c

100 corresponde ao caractere d

101 corresponde ao caractere e

102 corresponde ao caractere f

103 corresponde ao caractere g

104 corresponde ao caractere h

105 corresponde ao caractere i

106 corresponde ao caractere j

107 corresponde ao caractere k

108 corresponde ao caractere l

109 corresponde ao caractere m

110 corresponde ao caractere n

111 corresponde ao caractere o

112 corresponde ao caractere p

113 corresponde ao caractere q

114 corresponde ao caractere r

115 corresponde ao caractere s

116 corresponde ao caractere t

117 corresponde ao caractere u

118 corresponde ao caractere v

119 corresponde ao caractere w

120 corresponde ao caractere x

121 corresponde ao caractere y

122 corresponde ao caractere z

123 corresponde ao caractere {

124 corresponde ao caractere |

125 corresponde ao caractere }

126 corresponde ao caractere ~

127 corresponde ao caractere ⌂

6 -

DEFINIÇÃO DO PROBLEMA

O problema se define em fazer um gráfico com caracteres para representar y=x e y=2x e y=0.5x. Deve-se usar a rotina w\_gotoxy(x,y) baseada na API do Windows e a coordenada no monitor são x = colunas e y = linhas, com origem (0,0) no canto superior esquerdo.

SOLUÇÃO

Foi adicionado uma variável y que substituiu uma variável x, mas antes da substituição houve a adição de uma expressão, esta é y resultado final de x menos x, assim o valor de y será igual à inversão de x. E no final, foi adicionado um getchar() para impedir que apareça uma mensagem no meio do gráfico.

VALIDAÇÃO

Para validar o programa foi analisado o gráfico antes e depois da mudança, assim é possível saber se foi invertido.

COMENTÁRIOS E CONCLUSÕES

O gráfico foi invertido conforme.

O único problema encontrado foi a aparição da última mensagem no meio do gráfico.

LISTAGENS

Código comentado:

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

int w\_gotoxy( const WORD x, const WORD y )

{

COORD xy;

xy.X = x;

xy.Y = y;

return SetConsoleCursorPosition( GetStdHandle( STD\_OUTPUT\_HANDLE ), xy );

}

int main(void)

{

system("cls");

// foi adicionado as variaveis x e y

float x, y;

// estrutura de repetiçao para adicionar +1 em x

for(x=2; x<80; x++)

{

// y ser invertido por começar com o ultimo numero -x

y= 81-x;

// na casa do y o x foi trocado por y

w\_gotoxy(x, 0.25\*y);

putchar(178);

}

for(x=2; x<25; x++)

{

y= 26-x;

w\_gotoxy(x,1.0\*y);

putchar(178);

}

for(x=2; x<25; x++)

{

y= 26-x;

w\_gotoxy(x,1.5\*y);

putchar(178);

}

//pede para que seja algo seja digitado, impedindo inicialmente que apareca a ultima mensagem no meio da impressao

getchar();

system("pause");

return 0;

}

Resultados:

▓▓

▓▓▓▓

▓ ▓▓▓▓

▓▓ ▓▓▓▓

▓▓ ▓▓▓▓

▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓▓▓▓

▓ ▓ ▓▓▓▓

▓ ▓▓▓▓

▓▓▓▓▓

▓▓▓▓ ▓

▓▓▓▓ ▓ ▓

▓

▓ ▓

▓ ▓

▓

▓ ▓

▓

▓

▓

▓

▓

▓

▓

▓

7 -

DEFINIÇÃO DO PROBLEMA

O problema pede que mostremos a diferença entre as funções getch() e getchar().

SOLUÇÃO

A solução foi usar as funções getch() e getchar() como valores de diferentes variáveis para comparar o funcionamento de ambas

A função getch() não necessita da tecla “Enter” após digitar o caractere e a função getchar() exige a tecla “Enter” após a digitação.

VALIDAÇÃO

O teste de verificação se baseia ao digitar os caracteres no teclado para cada função, estas executem suas respectivas ações.

COMENTÁRIOS E CONCLUSÕES

Cada função agiu como era esperado na teoria

LISTAGENS

Código comentado:

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

int main()

{

char a, b;

// pede para digitar um caractere

printf("digite uma letra\n");

// recebe um caracere e escaneia

a= getchar();

// imprime o caractere escaneado, explica como o getchar funciona e pede para digitar um novo caractere

printf("Perceba que o caractere %c foi lido apos apertar enter, agora digite outra letra.\n", a);

// recebe um caracere e escaneia

b= getch();

// imprime o caractere escaneado e explica como o getch funciona

printf("\nPerceba que o caractere %c nao apareceu na tela e foi lido sem a necessidade do enter.\n", b);

printf("fim.\n");

system("pause");

return 0;

}

Resultados, saída:

digite uma letra

a

Perceba que o caractere a foi lido apos apertar enter, agora digite outra letra.

Perceba que o caractere b nao apareceu na tela e foi lido sem a necessidade do enter.

fim.

Pressione qualquer tecla para continuar. . .

8 -

DEFINIÇÃO DO PROBLEMA

O problema se baseia em descobrir se as expressões são falas(0) ou verdadeiras(1).

SOLUÇÃO

Para solucionar usamos os comandos if..else, pois estes impõem uma condição e assim facilita a resolução da expressão dada.

COMENTÁRIOS E CONCLUSÕES

Os resultados atingiram as expectativas.

Após a execução do código, checamos os resultados para ver se estão de acordo com teoria.

LISTAGENS

- Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main()

{

int a, c, b, d, e, f, g;

a= 1>2;

b= 'a'>'b';

c= 2==0;

d= '1'=='1';

e= 3>=2;

f= 'a'!= 'a';

g= a=0;

// confere se a expressao é verdadeira ou falsa

if(1>2)

{

//resultado se verdadeiro

printf("1>2 == %d entao verdadeiro\n", a);

}

else

{

// resultado se falso

printf("1>2 == %d entao falso\n", a);

}

if('a'>'b')

{

printf("'a'>'b' == %d entao verdadeiro\n", b);

}

else

{

printf("'a'>'b' == %d entao falso\n", b);

}

if(2==0)

{

printf("2==0 == %d entao verdadeiro\n", c);

}

else

{

printf("2==0 == %d entao falso\n", c);

}

if('1'=='1')

{

printf("'1'=='1' == %d entao verdadeiro\n", d);

}

else

{

printf("'1'=='1' == %d entao falso\n", d);

}

if(3>=2)

{

printf("3>=2 == %d entao verdadeiro\n", e);

}

else

{

printf("3>=2 == %d entao falso\n", e);

}

if('a'!= 'a')

{

printf("'1!= '1 == %d entao verdadeiro\n", f);

}

else

{

printf("'1'!= '1' == %d entao falso\n", f);

}

if(a=0)

{

printf("a=0 == %d entao verdadeiro\n", g);

}

else

{

printf("a=0 == %d entao falso\n", g);

}

system("pause");

return 0;

}

Resultado:

1>2 == 0 entao falso

'a'>'b' == 0 entao falso

2==0 == 0 entao falso

'1'=='1' == 1 entao verdadeiro

3>=2 == 1 entao verdadeiro

'1'!= '1' == 0 entao falso

a=0 == 0 entao falso

9 -

DEFINIÇÃO DO PROBLEMA

- O programa se define em escrever um programa que leia o número de horas semanais trabalhadas e o valor da hora de trabalho e calcular o salário semanal incluindo as horas extras pagando 50% a mais que o valor das horas normais e imprimir o salário do empregado considerando que até 40 horas trabalhadas na semana são consideradas horas normais, além disso a diferença é considerada como horas extras.

SOLUÇÃO

Para solucionar o problema, usamos expressões para calcular o salário semanal com e sem hora extra dentro do comando de condição if..else, pois assim poderíamos saber se o usuário trabalhou mais ou menos de 40 horas.

COMENTÁRIOS E CONCLUSÕES

Os resultados saíram de acordo com o esperado na teoria.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main()

{

int hs;

double vh, ss, ssh, he;

// pede para o usuario as horas trabalhadas

printf("Digite as horas semanais\n");

// colhe o resultado digitdo

scanf("%d", &hs);

// pede para o usuario o salario

printf("Agora o valor hora\n");

// colhe o resultado digitado

scanf("%lf", &vh);

// separa a formula que vai ser usada caso o horario seja menos ou mais que 40h semanais

if(hs < 41)

{

ss= hs\*vh;

// imprime o salario sem o extra, pois nao houve hora extra

printf("\nSalario semanal sem hora extra = %.2lf\n", ss);

}

else

{

ss= (((hs-40)\*(vh\*1.5))+(40\*vh));

// imprime o salario com o extra

printf("\nSalario semanal + hora extra = %.2lf\n", ss);

}

system("pause");

return 0;

}

Resultados:

Digite as horas semanais

40

Agora o valor hora

10

Salario semanal sem hora extra = 400.00

Pressione qualquer tecla para continuar. . .

10 -

DEFINIÇÃO DO PROBLEMA

O problema se define em ler os valores a, b e c dos coeficientes de uma equação do segundo grau, calcular e imprima as suas raízes. O programa deve tratar casos de raízes complexas, raízes reais e iguais, e raízes reais e diferentes. E tratar o caso em que a=0.

SOLUÇÃO

Para solucionar o problema abrimos uma variável(e) que tem como expressão, delta da fórmula de Báskara, após usamos o comando if...else, assim podemos estabelecer se o delta será igual a zero, maior que zero ou menor que zero.

VALIDAÇÃO

Para obter a validação, a impressão do código deve apresentar as mesmas raízes calculadas anteriormente de forma manual.

COMENTÁRIOS E CONCLUSÕES

O resultado está de acordo com o esperado.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

int main()

{

int a, b, c;

double e, f, g;

// pede para que seja digitado os 3 numeros da equaçao

printf("digite 3 numeros inteiros para a solucao da equacao do 2 grau, comecando do primeiro numero 'a'\n");

scanf("%d", &a);

printf("agora o segundo numero 'b'\n");

scanf("%d", &b);

printf("agora o ultimo numero 'c'\n");

scanf("%d", &c);

// formula do delta

e= (b\*b)-(4\*a\*c);

// separa as formulas caso delta=0 ou delta>0

if(e>=0)

{

// caso delta seja igual a 0

if(e==0)

{

// primeiro resultado

f= (-b+sqrt(e))/2\*a;

//segundo resultado

g= (-b-sqrt(e))/2\*a;

// impressao dos resultados

printf("\nAs duas raizes tem o mesmo valor sendo x1 = %.2f e x2 = %.2f.\n\n", f, g);

}

// caso delta seja maior que 0

else

{

f= (-b+sqrt(e))/2\*a;

g= (-b-sqrt(e))/2\*a;

printf("\nAs raizes tem o valor de x1 = %.2f e x2 = %.2f.\n\n", f, g);

}

}

// caso delta seja menor que 0

else

{

printf("\nA equacao nao possui raiz.\n\n");

}

system("pause");

return 0;

}

Resultados:

digite 3 numeros inteiros para a solucao da equacao do 2 grau, comecando do primeiro numero 'a'

1

agora o segundo numero 'b'

3

agora o ultimo numero 'c'

-4

As raizes tem o valor de x1 = 1.00 e x2 = -4.00.

Pressione qualquer tecla para continuar. . .

11 -

DEFINIÇÃO DO PROBLEMA

O problema se define em implementar uma calculadora de operações (\*, /, %, +, -) com números inteiros.

SOLUÇÃO

Para solucionar, usamos o comando scanf() para pedir ao usuário que digite os dois números e o caractere de operação, em seguida usamos o comando switch...case, assim podemos declarar dentro de cada case o que cada operação faz.

VALIDAÇÃO

Para obter a validação, o programa deve solucionar as expressões de acordo com a respectiva operação.

COMENTÁRIOS E CONCLUSÕES

O resultado está de acordo com a teoria.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main()

{

int a, b;

char c;

double d;

// pede para o usuario digitar uma expressao

printf("digite a soma\n");

// escaneia a expressao digitada

scanf("%d %c %d", &a, &c, &b);

// separa a funcao caso a expressao seja soma, subtração, multiplicaçao, divisao ou resto

switch (c)

{

// caso a expressao seja uma soma

case '+':

{

d= a+b;

// impressao do resultado caso seja uma soma

printf("resultado da soma = %.2f\n", d);

// saida da funçao switch

break;

}

// caso a expressao seja uma subtraçao

case '-':

{

d= a-b;

// impressao do resultado caso seja uma subtraçao

printf("resultado da soma = %.2f\n", d);

// saida da funçao switch

break;

}

case '\*':

{

d= a\*b;

printf("resultado da soma = %.2f\n", d);

break;

}

case '/':

{

d= a/b;

printf("resultado da soma = %.2f\n", d);

break;

}

case '%':

{

d= a%b;

printf("resultado da soma = %.5f\n", d);

break;

}

}

system("pause");

return 0;

}

Resultados:

digite a soma

1+3

resultado da soma = 4.00

Pressione qualquer tecla para continuar. . .

digite a soma

1-3

resultado da soma = -2.00

Pressione qualquer tecla para continuar. . .

digite a soma

1\*3

resultado da soma = 3.00

Pressione qualquer tecla para continuar. . .

digite a soma

6/3

resultado da soma = 2.00

Pressione qualquer tecla para continuar. . .

digite a soma

10%3

resultado da soma = 1.00000

Pressione qualquer tecla para continuar. . .

12 -

DEFINIÇÃO DO PROBLEMA

O problema é se baseia em ler dois números inteiros positivos, n\_1 e n\_2, e calcule a soma dos múltiplos de 3 entre eles.

SOLUÇÃO

A solução do problema foi criar um código que lê 2 valores, foi utilizado uma estrutura de repetição e outra de condição, assim era possível somar o número em uma variável caso fosse múltiplo de 3, após isso, imprimir a soma dos números múltiplos de 3.

VALIDAÇÃO

O teste foi feito com a execução do programa utilizando valores numéricos pequenos, assim pode-se ter mais precisão.

COMENTÁRIOS E CONCLUSÕES

A conclusão foi de acordo com o esperado.

LISTAGENS

Código comentado:

#include<stdio.h>

#include<stdlib.h>

int main()

{

int a, b, c, d=0;

// pede para o usuario 2 numeros separados

printf("Digite um numero.\n");

scanf("%d", &a);

printf("Agora outro numero.\n");

scanf("%d", &b);

//separa qual funcao sera utilizada

if(a<b)

{

// estrutura de repetiçao para adicionar +1 em c ate c=b

for(c=a; c<=b; c++)

{

// verifica se c é multiplo de 3

if(c%3==0)

{

// soma +c em d

d= d+c;

}

}

// imprime a soma dos multiplos de 3 apos o termino da estrutura de repetiçao

printf("Soma dos numeros multiplos de 3 = %d.\n", d);

}

// caso a>b

else

{

// estrutura de repetiçao para adicionar +1 em c ate c=a

for(c=b; c<=a; c++)

{

if(c%3==0)

{

d= d+c;

}

}

printf("Soma dos numeros multiplos de 3 = %d.\n", d);

}

system("pause");

return 0;

}

Resultados:

Digite um numero.

1

Agora outro numero.

99

Soma dos numeros multiplos de 3 = 1683.

Pressione qualquer tecla para continuar. . .