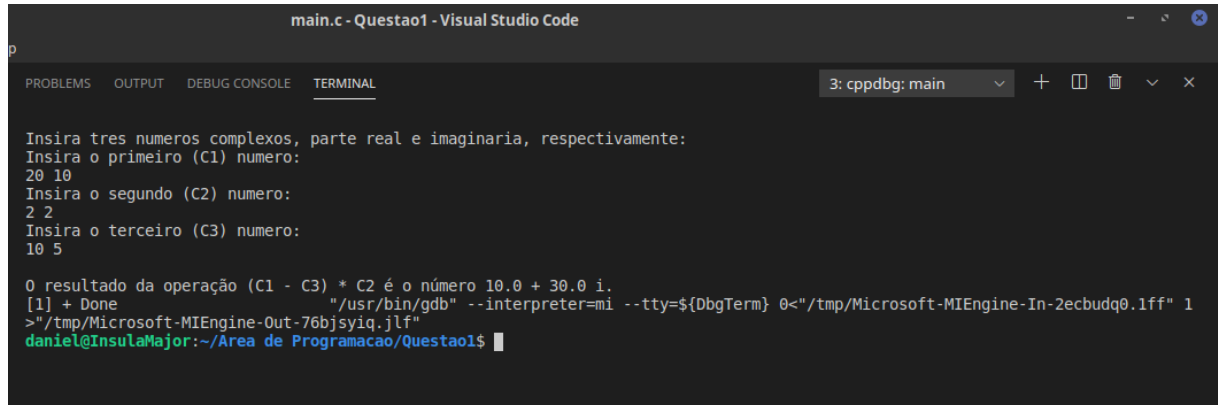


## Prática Avaliativa – Tipo Abstrato de Dados


Daniel Gonçalves  
12011BCC011

### Questão 1:



```
main.c - Questao1 - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3: cppdbg: main
Insira tres numeros complexos, parte real e imaginaria, respectivamente:
Insira o primeiro (C1) numero:
20 10
Insira o segundo (C2) numero:
2 2
Insira o terceiro (C3) numero:
10 5

O resultado da operação (C1 - C3) * C2 é o número 10.0 + 30.0 i.
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-2ecbudq0.1ff" 1
>"/tmp/Microsoft-MIEngine-Out-76bjsyiq.jlf"
daniel@InsulaMajor:~/Area de Programacao/Questao1$
```



```
main.c - Questao1 - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3: cppdbg: main
Insira tres numeros complexos, parte real e imaginaria, respectivamente:
Insira o primeiro (C1) numero:
10 5
Insira o segundo (C2) numero:
0 0
Insira o terceiro (C3) numero:
12 2

O resultado da operação (C1 - C3) * C2 é o número 0.0 + 0.0 i.
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-4z1h8s5p.zef" 1
>"/tmp/Microsoft-MIEngine-Out-rr95ixg3.vta"
daniel@InsulaMajor:~/Area de Programacao/Questao1$
```

Nome TAD: complexo

Dados: double, representando a parte real; double, representando a parte imaginária.

Lista de operações: criar\_nro, libera\_nro, soma, sub, mult.

Operações:

criar\_nro:

Entrada: dois doubles, um para a parte real e outro para a imaginaria

Pré-condição: nenhuma

Processo: aloca uma estrutura do tipo complexo e atribui os valores dados.

Saída: endereço para estrutura do tipo complexo, em sucesso, NULL, em fracasso

Pós-condição: os campos da estrutura foram preenchidos com os valores passados

libera\_nro:

Entrada: ponteiro para ponteiro do tipo complexo

Pré-condição: nenhuma

Processo: desaloca a estrutura

Saída: nenhuma

Pós-condição: a estrutura foi desalocada.

soma:

Entrada: dois endereços para estruturas do tipo complexo

Pré-condição: ambos endereços precisam ser válidos.

Processo: aloca uma terceira estrutura, realiza a soma e atribui o resultado a essa nova estrutura.

Saída: o endereço da terceira estrutura, em sucesso, NULL, em fracasso

Pós-condição: nenhuma

sub:

Entrada: dois endereços para estruturas do tipo complexo

Pré-condição: ambos endereços precisam ser válidos.

Processo: aloca uma terceira estrutura, realiza a subtração e atribui o resultado a essa nova estrutura.

Saída: o endereço da terceira estrutura, em sucesso, NULL, em fracasso

Pós-condição: nenhuma

mult:

Entrada: dois endereços para estruturas do tipo complexo

Pré-condição: ambos endereços precisam ser válidos.

Processo: aloca uma terceira estrutura, realiza a multiplicação e atribui o resultado a essa nova estrutura.

Saída: o endereço da terceira estrutura, em sucesso, NULL, em fracasso

Pós-condição: nenhuma,

## Questao 2:

```
main.c - Questao2 - Visual Studio Code
elp
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3: cppdbg: main
----Entre com as coordenadas para dois pontos tridimensionais:----
Primeiro ponto, coordenadas x,y e z, respectivamente:
0 0 0
Segundo ponto, coordenadas x,y e z, respectivamente:
1 1 1
A distancia entre os pontos inseridos é de 1.73
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-kijrlhpt.qc1" 1
>"/tmp/Microsoft-MIEngine-Out-gam4tnd7.j7a"
daniel@InsulaMajor:~/Area de Programacao/Questao2$
```

```
main.c - Questao2 - Visual Studio Code
elp
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3: cppdbg: main
----Entre com as coordenadas para dois pontos tridimensionais:----
Primeiro ponto, coordenadas x,y e z, respectivamente:
0 0 0
Segundo ponto, coordenadas x,y e z, respectivamente:
0 0 1
A distancia entre os pontos inseridos é de 1.00
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-0q08b00f.6wy" 1
>"/tmp/Microsoft-MIEngine-Out-0gpnffib.a9w"
daniel@InsulaMajor:~/Area de Programacao/Questao2$
```

```
main.c - Questao2 - Visual Studio Code
elp
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3: cppdbg: main
----Entre com as coordenadas para dois pontos tridimensionais:----
Primeiro ponto, coordenadas x,y e z, respectivamente:
0 0 0
Segundo ponto, coordenadas x,y e z, respectivamente:
1 1 0
A distancia entre os pontos inseridos é de 1.41
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-zi8qzefk.wlh" 1
>"/tmp/Microsoft-MIEngine-Out-2btp6ywa.qsn"
daniel@InsulaMajor:~/Area de Programacao/Questao2$
```

Nome TAD: ponto\_tridimensional

Dados: três inteiros, cada um representando as coordenadas x,y,z.

Lista de operações: criar\_ponto, preenche\_ponto, libera\_ponto, distancia\_pontos

Operações:

criar\_ponto:

Entrada: três inteiros, cada um representando uma das coordenadas, x, y e z, respectivamente.

Pré-condição: nenhuma

Processo: aloca dinamicamente uma estrutura do tipo ponto\_tri e preenche seus campos com os dados recebidos

Saída: endereço para estrutura do tipo ponto\_tri, em sucesso, NULL, em fracasso

Pós-condição: nenhuma

libera\_ponto:

Entrada: ponteiro para ponteiro para uma estrutura do tipo ponto\_tri.

Pré-condição: nenhuma  
Processo: desaloca a estrutura apontada pelo ponteiro recebido e atribui NULL a ele.  
Saída: nenhuma  
Pós-condição: estrutura desaloca e ponteiro apontando para NULL.

distancia\_pontos:

Entrada: ponteiro para duas estruturas do tipo ponto\_tri.  
Pré-condição: ambos ponteiros precisam apontar para endereços válidos.  
Processo: calcula a distancia euclidiana entre os pontos recebidos  
Saída: -1, em falha, a distancia euclidiana, em sucesso. Saída em double  
Pós-condição: nenhuma.

get\_x:

Entrada: ponteiro para uma estrutura do tipo ponto\_tri.  
Pré-condição: o ponteiro deve apontar para um endereço válido. (Essa verificação deve ser feita antes da função ser chamada, pois não existe jeito de se retornar um flag avisando de erro).  
Processo: acessa o campo da estrutura especificado  
Saída: a coordenada x.  
Pós-condição: nenhuma

get\_y:

Entrada: ponteiro para uma estrutura do tipo ponto\_tri.  
Pré-condição: o ponteiro deve apontar para um endereço válido. (Essa verificação deve ser feita antes da função ser chamada, pois não existe jeito de se retornar um flag avisando de erro).  
Processo: acessa o campo da estrutura especificado  
Saída: a coordenada y.  
Pós-condição: nenhuma

get\_z:

Entrada: ponteiro para uma estrutura do tipo ponto\_tri.  
Pré-condição: o ponteiro deve apontar para um endereço válido. (Essa verificação deve ser feita antes da função ser chamada, pois não existe jeito de se retornar um flag avisando de erro).  
Processo: acessa o campo da estrutura especificado  
Saída: a coordenada z.  
Pós-condição: nenhuma

### Questão 3:

```
main.c - Questao3 - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: cppdbg: main
Entre com as coordenadas x,y e z do ponto que representa o centro da esfera:
0 0 0
Entre com as coordenadas x,y e z do ponto que representa a superfície da esfera:
0 0 1
---A esfera passada possui:---
raio = 1.00
area = 12.57
volume = 4.19
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-3oh3pbk4.igx" 1
>"/tmp/Microsoft-MIEngine-Out-ijtnxheh.hgv"
daniel@InsulaMajor:~/Area de Programacao/Questao3$
```

```
main.c - Questao3 - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: cppdbg: main
Entre com as coordenadas x,y e z do ponto que representa o centro da esfera:
0 0 0
Entre com as coordenadas x,y e z do ponto que representa a superfície da esfera:
2 0 0
---A esfera passada possui:---
raio = 2.00
area = 50.26
volume = 33.51
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-v9fpxduo.qsv" 1
>"/tmp/Microsoft-MIEngine-Out-0m9qpuz7.bjh"
daniel@InsulaMajor:~/Area de Programacao/Questao3$
```

```
main.c - Questao3 - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: cppdbg: main
Entre com as coordenadas x,y e z do ponto que representa o centro da esfera:
0 0 0
Entre com as coordenadas x,y e z do ponto que representa a superfície da esfera:
0 0 0
Falha! -- Entre com os dados novamente!
Entre com as coordenadas x,y e z do ponto que representa o centro da esfera:
0 0 0
Entre com as coordenadas x,y e z do ponto que representa a superfície da esfera:
1 1 1
---A esfera passada possui:---
raio = 1.73
area = 37.70
volume = 21.76
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-ekxg2mzf.ctm" 1
>"/tmp/Microsoft-MIEngine-Out-l2h6mwlo.o5w"
daniel@InsulaMajor:~/Area de Programacao/Questao3$
```

Nome TAD: esfera

Dados: duas estruturas do tipo ponto\_tri, representando o centro da esfera e um ponto qualquer na superfície da esfera. O raio é a distancia entre esses dois pontos.

Lista de operações: criar\_esfera, libera\_esfera, raio, area, volume

Operações:

criar\_esfera:

Entrada: seis inteiros, representando as coordenadas de dois pontos, respectivamente: x1, y1, z1; x2, y2, z2

Pré-condição: os pontos não podem ser iguais

Processo: aloca dinamicamente uma estrutura do tipo Esfera e preenche seus campos.

Saída: ponteiro para estrutura do tipo Esfera, em sucesso, NULL, em fracasso.

Pós-condição: nenhuma

libera\_esfera:

Entrada: ponteiro para ponteiro para estrutura do tipo esfera

Pré-condição: nenhuma

Processo: desaloca a estrutura apontada e atribui NULL ao ponteiro

Saída: nenhuma

Pós-condição: estrutura desaloca e ponteiro apontando para NULL

raio\_esfera:

Entrada: ponteiro para estrutura do tipo Esfera

Pré-condição: o ponteiro precisa apontar para um endereço válido

Processo: aloca duas estruturas do tipo ponto\_tri e preenche elas com os dados retirados da estrutura do tipo esfera e

então chama a função que calcula distancia entre pontos.

Saída: 0, em falha, ou um inteiro não nulo, representando o raio da esfera.

Pós-condição: nenhuma

area\_esfera:

Entrada: inteiro, representando o raio de uma esfera

Pré-condição: o valor do raio precisa ser maior que zero

Processo: calcula a area superficial de uma esfera de raio que foi passado

Saída: 0, em falha, um double representando o raio

Pós-condição: nenhuma

volume\_esfera:

Entrada: inteiro, representando o raio de uma esfera

Pré-condição: o valor do raio precisa ser maior que zero

Processo: calcula o volume de uma esfera com o raio passado

Saída: 0, em falha, um double representando o raio, em sucesso

Pós-condição: nenhuma