Lista 1 – PP

O objetivo desta lista é auxiliar os alunos a fixar o conteúdo e ampliar seus conhecimentos. Não vale nota, porém é recomendado que a façam.

Para os exercícios desta lista, implemente geters, seters, contrutores e destrutores sempre que necessário. Escolha com cuidado os dados membros (não use dados desnecessários e sempre inclua-os na sessão privada da classe), tipos de retorno e parâmetros dos métodos. Implemente uma aplicação simples para testar as funcionalidades implementadas (não precisa realizar entrada pelo usuário, apenas crie objetos, chame os métodos implementados e imprima as saídas resultantes). Não utilize entrada e saída dentro de classe que não são especificar para isso.

Vamos começar com um exercício básico:

1- Escreva uma classe para representar um par de inteiros (first, second). Implemente métodos para:
a- trocar o valore armazenados em first pelo valor de second e vice-versa (swap);
b- retornar o maior valor entre first e second.

O próximo exercício utiliza duas classes e conhecimentos de geometria.

2- Implemente uma classe para representar pontos em R^2 (vulgo 2D) e implemente um método para:
a- calcular a distância entre dois pontos (será usado na próxima parte do exercício).

Agora, usando a classe acima, implemente uma segunda classe para representação de triângulos, descritos por três pontos (vértices do triângulo). Implemente métodos para:
a- calcular a área do triângulo;
b- verificar a classificação deste triângulo quanto ao tamanho de seus lados (isósceles, equilátero e escaleno)
c- verificar a classificação deste triângulo quanto aos seus ângulos internos (acutângulo, obtusângulo e retângulo);

Obs ex 2: utilize uma enumeração para os tipos de triângulo em relação aos lados e outra em relação aos ângulos (pesquise “C++ enum” se não sabe como fazê-lo). Enumerações tornam o código mais legível.

Agora, um exercício com matrizes.

3- Implemente uma classe para representar matrizes, de ponto flutuante, 2X2. Implemente métodos para:
a- calcular o determinante;
b- soma de duas matrizes;
c- multiplicação de duas matrizes;
d- calculo de matriz inversa.

O próximo exercício é uma estrutura de dados.

4- Implemente uma classe para representar arranjo de inteiros ordenados que comporte no máximo 10 elementos. Implemente funções para:
a- busca um elemento no array (implemente usando busca binária)

b- inserir um elemento (inserção ordenada). (Opcional, mas recomendado) Pesquise o tema lançamento de exceção em C++. Quando o arranjo estiver cheio, lance uma exceção.
c- remover o item por valor. Nesse caso, o método deve retornar um booleano: true quando o elemento for encontrado e false caso contrário(Opcional, mas recomendado) lance uma exceção quando a arranjo estiver vazio.

Agora, vamos escovar bit´s.

5- Implemente uma classe capaz de armazenar oito bits. O único dado membro permitido é uma variável do tipo byte (ou char se preferir, pois são análogas). Implemente métodos para:
a- um construtor capaz de receber uma cadeia de caracteres (ou string) contendo uma sequencia de 8 0’s e 1’s;
b- que retorne o i-ésimo bit (booleano – bool);
c- que modifique o i-ésimo bit (booleano – bool);
d- que retorne a paridade do byte (par ou impar de acordo com número de 1’s nesse byte). Use uma enumeração para este método (vide ex 2).

Usando alocação dinâmica:

6- Idem ao ex 4, para um número qualquer de elementos. Adote a política de dobrar o arranjo quando, na tentativa de inserir um elemento, o arranjo estiver cheio. Reduzir pela metade o tamanho do arranjo, quando na remoção, tivermos metade do arranjo sem elementos.