UNIVERSIDADE FEDERAL DE SANTA CATARINA

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE - CTS
DEPARTAMENTO DE COMPUTAÇÃO – DEC

DISCIPLINA: LINGUAGENS DE PROGRAMAÇÃO 2
PROFESSOR ANTONIO CARLOS SOBIERANSKI
a.sobieranski@ufsc.br

ENUNCIADO TRABALHO T2 – CIPHER

A ser desenvolvido individualmente.

O cifrador de texto desenvolvido anteriormente por tabela de substituição (*Cipher.cpp*) deve ser convertido para sua versão orientada a objetos.

Converta a solução levando em consideração:

1. Criar a classe *ConversionTable*, responsável pelos aspectos da tabela de conversão e métodos.
2. Criar a classe *Cipher*, responsável por encriptar e decriptar textos. Esta classe possui um objeto da classe *ConversionTable* como atributo, que deve ser criado no construtor de *Cipher* **dinamicamente** e estar encapsulado. Ninguém deve ter acesso à tabela de conversão.
3. A classe *Cipher* deve se comunicar com seu atributo do tipo *ConversionTable*, criando-o de alguma forma que seu código de aluno seja informado também, como uma espécie de chave a ser combinada na estratégia de geração da tabela de conversão. Logo, o código de aluno deve vir do main para o objeto de *Cipher*, e ser passado de *Cipher* para a *ConvertionTable*.
4. Criar em *Cipher* um método para re-gerar a tabela de conversão, sempre que for solicitado (não somente no construtor), inclusive passando outro código de aluno como chave se for necessário a partir do main.
5. O main se comunicará com um objeto do tipo *Cipher*, enviando textos a serem encriptados, e decriptados para o objeto resolver.
6. A estratégia para gerar a tabela de conversão é livre, e pode envolver outras estratégias de sua escolha, desde que usada combinada com seu código de aluno. Lembre-se que deve ser reversível para poder decriptar os textos.
7. Elaborar um menu.

Entregável no Moodle:

- Código fonte, considerando somente *cpp's* e *hpp's* (ou .h) do programa desenvolvido. Deve compilar e ser possível de testes.
- Vídeo de no máximo 5 minutos (ou link permanente) demonstrando a estratégia utilizada no código, e sua execução.

Dicas:
Alocar memória e desalocar apropriadamente com *new* e *delete*
Alguém disse: Debug salva vidas….