UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
ESCOLA DE CIÊNCIAS E TECNOLOGIA

Projeto Final

Professores: Diego Arthur de Azevedo Morais, Einstein Gomes dos Santos

1. OBJETIVO

O objetivo deste trabalho é a implementação de uma agenda de contatos. Durante a implementação da agenda, os alunos irão desenvolver suas competências no uso de estruturas de repetição, vetores, strings, structs, funções, procedimentos, entrada e saída de dados e programação estruturada.

2. PROJETO

O projeto deverá ser desenvolvido na linguagem de programação C++.

O trabalho deverá ser desenvolvido individualmente. Qualquer aluno que não entregar o trabalho no prazo, receberá nota 0.

Ao término do projeto, deverá ser enviado para o SIGAA um arquivo compactado (extensão .zip), contendo:

- Código (arquivo.cpp)
- Relatório (arquivo.pdf)

A formatação do relatório é livre, e o relatório deverá conter:

- Nome e subturma
- Evolução do projeto
  - Detalhar o tempo gasto, se o prazo foi curto, ou extenso, se a dedicação foi constante, ou não, ao longo do tempo
- Funcionalidades que não foram concluídas
  - Caso alguma funcionalidade não seja concluída, devem ser especificados os motivos (dificuldades na implementação de alguma funcionalidade, falta de base teórica ou de experiência prática, erros de tempo de compilação, erros de tempo de execução)
- Funcionalidades concluídas
  - Deve ser especificada a lógica utilizada na implementação, o código responsável por aquela funcionalidade e se houve ajuda na implementação (se houve ajuda do professor, monitor ou amigos)
- Lições aprendidas por cada integrante
  - Detalhar qualquer coisa aprendida durante a implementação
- Comentários gerais

    - Espaço livre para comentários

3. CRONOGRAMA

O projeto será desenvolvido em 3 etapas.

Etapa 1: 18/11

- Planejamento
    - Criação da struct e protótipos das funções e dos procedimentos

Etapa 2: 25/11

- Desenvolvimento das funcionalidades
    - Implementação das funções e procedimentos

Etapa 3: 02/12

- Ajustes finais e testes
    - Correções e verificações

Data limite da entrega do projeto final: 02/12

As datas das 3 etapas coincidem com as datas das nossas aulas de laboratório. Nestas datas, todos os alunos devem comparecer à aula, mesmo que tenham adiantado o projeto em casa.

O comparecimento é obrigatório, qualquer aluno que não comparecer será penalizado, sendo a penalidade máxima a não correção do projeto. O comparecimento se faz necessário para a orientação no desenvolvimento das etapas, e acompanhamento do que foi feito em casa.

4. DÚVIDAS

Dúvidas deverão ser tiradas com os monitores (nos horários estabelecidos e disponibilizados pelos mesmos), ou com os professores.

Professor Diego: Caso desejem entrar em contato comigo, peço que utilizem o meu e-mail pessoal (arthur.vinx@gmail.com), responderei na medida do possível.

Lembrando que os professores e monitores estarão no laboratório (exceto no horário M12, que não tem monitor; no horário M34 o monitor Jetterson Lucas pode comparecer; no horário M56 sempre temos a monitora Vanessa Dantas; e no horário T12, sempre temos o monitor Yuri Pinheiro), nos dias 18/11, 25/11 e 02/12 para acompanhar o desenvolvimento do projeto.

5. NOTA

Este projeto será o único responsável pela nota da 3ª unidade do laboratório, valendo 100% da nota da unidade, cujo valor máximo é de 3 pontos na média da unidade.

Observação: Caso plágio seja detectado, os trabalhos “clones”, bem como o original, receberão nota 0.

A nota será composta pelo conteúdo do relatório e do código entregue (levando em conta as penalidades de não comparecimento). As funcionalidades a serem implementadas são:

- Exibir menu
- Inserir contato
- Buscar contato
    - Por nome
    - Por telefone
    - Por e-mail
- Atualizar contato
    - O contato deve existir para que possa ser atualizado
- Remover contato
    - Por nome
    - Por telefone
    - Por e-mail

6. DESCRIÇÃO DA AGENDA

Sua agenda usará uma Estrutura denominada Contato, esta estrutura possui 3 membros: uma string nome (tamanho máximo 50), uma string telefone (tamanho máximo 20, para os casos como +55 (84) 91234-5678), e uma string email (tamanho máximo 50).

Contato:

- Nome
- Telefone
- Email

Na função principal, você deverá criar um vetor denominado Agenda, capaz de armazenar 100 contatos.

Exemplo do programa:

- Deve ser exibido um menu com as opções disponíveis
    - 1 – Inserir contato
      2 – Buscar contato por nome
      3 – Buscar contato por telefone
      ...
      X – Sair
    - Lembre-se que ao sair, todas as informações serão perdidas, então crie linhas de código para povoar sua agenda com informações pré-definidas, deste modo, será mais fácil testar o que for feito
    - Lembre-se que um número de telefone pode ser digitado em diversos formatos, você deve tratar isto
    - Caso um número inválido seja informado, o menu deve ser exibido novamente
- Um contato só poderá ser adicionado caso a agenda não esteja lotada
    - O usuário pode optar por adicionar apenas o telefone
    - Você só deve adicionar um contato caso ele não exista
- Buscar contato
    - O contato pode existir na agenda, a agenda pode estar vazia ou o contato pode não existir
- Atualizar contato
    - Para atualizar um contato, ele deve existir

    - Caso não exista, o usuário deve ser informado
    - Caso exista, o usuário pode alterar qualquer informação do contato
- Remover contato
    - Para remover um contato, ele deve existir
    - Caso exista, o usuário o remove
    - Caso não exista, o usuário deve ser informado

7. DICAS ÚTEIS PARA A IMPLEMENTAÇÃO

Organize a saída do seu programa. Use a função system(“clear”) para limpar a tela do prompt do Linux, ou system(“cls”) para limpar a tela do prompt do Windows. Ambos os comandos são da biblioteca <cstdlib>.

Use acentos nas saídas com a função setlocale():

Exemplo:
```
int main(){
setlocale(LC_ALL, “Portuguese”)
cout<<”Olá”;
```

8. DICAS OPCIONAIS

Caso deseje pausar o programa por alguns segundos, use a função Sleep(N) no Windows, ou sleep(N) no Linux, respectivamente das bibliotecas <windows.h> e <unistd.h>. O valor de N é dado em milissegundos, após o período de pausa, o código seguirá normalmente.

Caso deseje pausar o programa até que o usuário pressione alguma tecla, use a função system(“pause”) da biblioteca <cstdlib>.

Caso deseje deixar o jogo mais atraente, use um gerador de arte ASCII. Basta copiar o texto gerado (coloque cada linha num cout<<LINHA<<endl;).

http://patorjk.com/software/taag/#p=display&h=1&v=1&f=Big&t=AGENDA

Não tente complicar as coisas, faça uma implementação simples, porém correta.

Pense antes de codificar, calma e planejamento são muito úteis.

Não deixe para fazer de última hora, você tem pouca coisa para fazer, mas muita coisa para pensar e tratar.