

TRABALHO PRÁTICO 3

Quando garoto, Joãozinho sempre jogava Super Trunfo com seus colegas no intervalo do colégio. Agora que ele aprendeu a programar, teve a ideia de criar um jogo de Super Trunfo para o computador. Ajude Joãozinho nessa tarefa.

Crie um jogo de **Super Trunfo** usando o tema de sua preferência (carros, aviões, esportes, etc). O Super Trunfo é um jogo de cartas em que cada carta tem um conjunto de atributos numéricos. Dois ou mais jogadores competem escolhendo atributos de uma carta e aquele que tem o maior atributo vence, levando as cartas dos oponentes. Quem conseguir ficar com todas as cartas do baralho vence.



Para o trabalho, o jogo será levemente adaptado para simplificar a sua programação:

- Existem apenas 2 competidores
- Cada competidor recebe 4 cartas aleatórias de um conjunto de cartas
- Competidor 1 e 2 se alternam na escolha de atributos em cada rodada
- A cada rodada um novo par de cartas é selecionado
- Cada vitória conta 2 pontos, empates contam 1 ponto
- Ao final de 4 rodadas, o competidor que tiver mais pontos ganha

O trabalho consiste na construção de dois programas separados:

- Gerador de Cartas
- Jogo

GERADOR DE CARTAS

O programa gerador de cartas deve apresentar um menu de opções que permita cadastrar, importar, alterar, excluir e listar as cartas do baralho:

Gerador de Cartas

(C)adastrar
(I)mportar
(A)lterar
(E)xcluir
(L)istar
(S)air

Escolha uma opção [_]

Cada opção deve permitir que o usuário entre com os dados necessários para completar aquela tarefa. Abaixo está uma sugestão de tela para cada opção e uma descrição do que deve ser feito para cada uma delas:

- **Cadastrar:** deve adicionar um novo registro ao vetor baralho. Este registro deve conter todos os dados de uma carta do jogo, como no exemplo abaixo:

Cadastrar Carta

Nome : **Messi**
Idade : **25**
Jogos : **418**
Gols : **301**
Títulos : **21**
Salário : **10.5**

- **Importar:** deve ler os dados das cartas a partir de um arquivo texto, cadastrando-as no vetor baralho como se fossem novos cadastros, sem remover os elementos já inseridos anteriormente. O arquivo texto deve conter uma carta por linha, com os atributos de cada carta separados por espaço.

Importar Cartas

Arquivo: **maiscartas.txt**

Importando:

Ronaldo 27 581 324 10
Kaka 30 576 194 13 9
Neymar 20 240 146 9 10.2

- **Alterar:** deve mostrar uma lista numerada das cartas existentes no baralho. Ao digitar um número, o usuário deve ser capaz de entrar com os dados atualizados daquela carta.

Atualizar Cartas

1) Messi
2) Ronaldo
3) Kaká
4) Neymar

Digite o número da carta: [2]

Alterando Carta Ronaldo:

Nome : **Ronaldo**
Idade : **27**
Jogos : **581**
Gols : **324**
Títulos : **12**
Salário : **10**

- **Excluir:** deve mostrar uma lista numerada das cartas existentes no baralho. Ao digitar um número, a carta deve ser excluída do baralho e não mais aparecer em futuras listagens.

Excluir Carta

- 1) Messi
- 2) Ronaldo
- 3) Kaká
- 4) Neymar

Digite o número da carta: [3]

Carta Kaká foi excluída do baralho.

- **Listar:** deve exibir todas as cartas do baralho com seus respectivos atributos.

Cartas no Baralho

#1	Messi	25	418	301	21	10.5
#2	Ronaldo	27	581	324	12	10.0
#3	Neymar	20	240	148	9	10.2

JOGO

O exemplo abaixo mostra a sequência de 4 rodadas que compõe uma partida do jogo. A cada partida deve ser perguntado ao usuário se ele quer iniciar uma nova partida ou sair do jogo.

Super Trunfo Futebol

Iniciar nova partida? [S/N] S

Jogador 1: **João**

Jogador 2: **José**

[João]

Carta: Neymar

1. Idade
2. Jogos
3. Gols
4. Títulos

5. Salário
Escolha atributo [3]

[João] Neymar | 240
[José] Kaká | 194

Placar: João 2 x 0 José

[José]

Carta: Pelé
1. Idade
2. Jogos
3. Gols
4. Títulos
5. Salário
Escolha atributo [1]

[José] Pelé | 60
[João] Messi | 25

Placar: João 2 x 2 José

[João]

Carta: Romário
1. Idade
2. Jogos
3. Gols
4. Títulos
5. Salário
Escolha atributo [2]

[João] Romário | 540
[José] Bebeto | 600

Placar: João 2 x 4 José

[José]

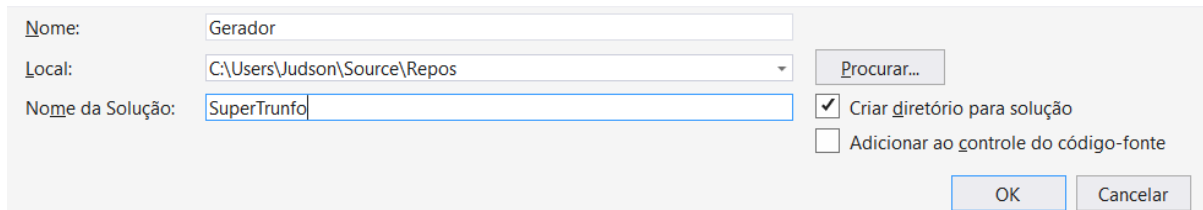
Carta: Ronaldo
1. Idade
2. Jogos
3. Gols
4. Títulos
5. Salário
Escolha atributo [5]

[José] Ronaldo | 10.0
[João] Zico | 10.0

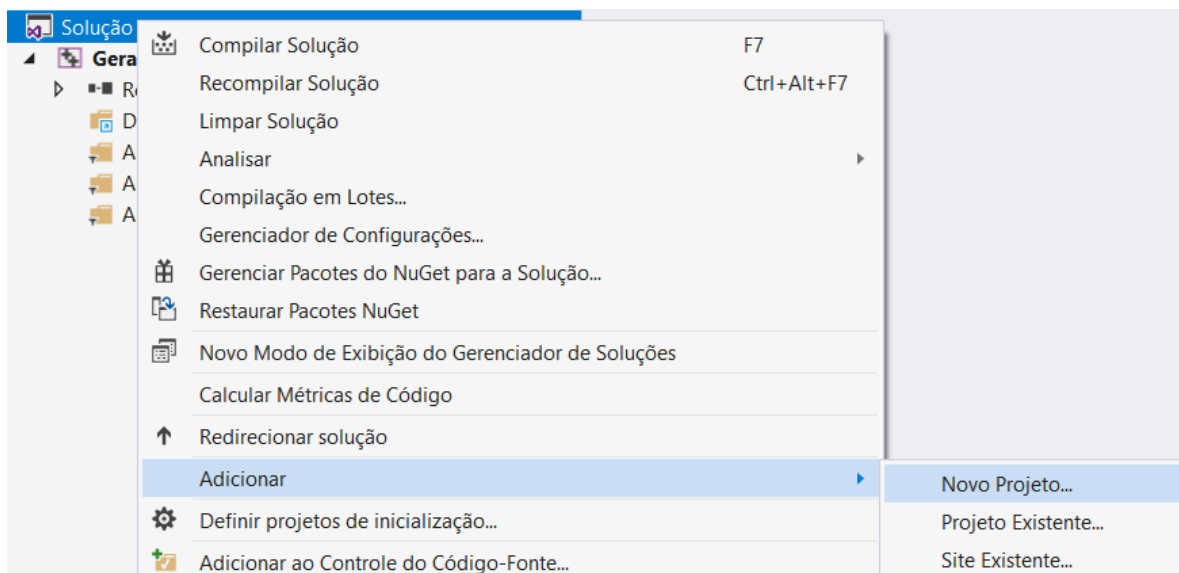
Final: João 3 x 5 José

CONFIGURAÇÃO DOS PROJETOS

No Visual Studio, os dois programas devem ser criados através de dois projetos dentro da mesma solução. Na criação do projeto, forneça um nome para o projeto e outro nome para a solução, como no exemplo abaixo.



Uma vez criados o primeiro projeto e a solução, adicione um segundo projeto na mesma solução clicando com o botão direito do mouse em Solução > Adicionar > Novo Projeto...

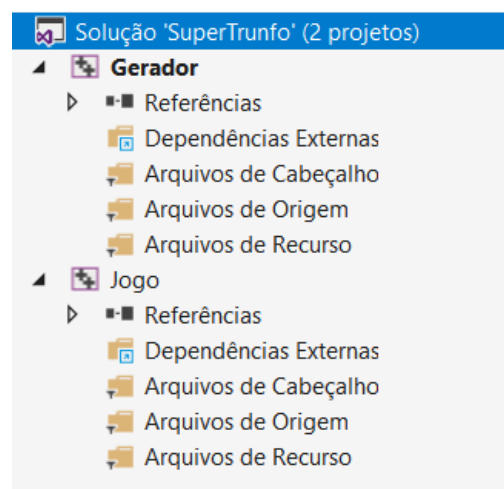


O projeto que foi adicionado primeiro será o **projeto de inicialização**. Ele fica com seu nome em negrito e é aquele que é executado ao se clicar no botão “Depurador Local do Windows”.

É possível mudar o projeto de inicialização a qualquer momento clicando com o botão direito do mouse no nome do projeto e escolhendo a opção “Definir como projeto de inicialização”.

Com essa organização, os programas podem compartilhar facilmente um arquivo salvando-o na pasta da solução. Para isso basta usar a notação “.. \” na hora de fornecer o nome do arquivo, como no exemplo abaixo:

```
fout.open(".. \baralho.dat");
```



INSTRUÇÕES

1. O “Gerador de Cartas” deve usar um vetor estático para armazenar até 32 cartas. O vetor deve armazenar temporariamente o baralho de cartas na memória a medida que as cartas vão sendo inseridas, removidas ou alteradas dentro do gerador.
2. O vetor deve ser preenchido com as cartas vindas de um **arquivo binário** toda vez que o usuário entrar no gerador. O **conteúdo válido do vetor** deve ser gravado **no mesmo arquivo binário** toda vez que o usuário sair do gerador.
3. O arquivo binário deve ser lido uma vez no início do programa e atualizado uma vez apenas no fim do programa. As inserções, remoções e alterações de cartas devem ser feitas no vetor e não diretamente no arquivo.
4. Em sua inicialização, o “Jogo” deve ler o arquivo binário de dados e passar as informações para um **vetor dinâmico** de cartas, cujo tamanho deve ser igual a quantidade de cartas armazenadas no arquivo. Uma vez lido, o arquivo deve ser fechado antes de iniciar a partida. **O arquivo não deve ser alterado pelo Jogo.**
5. O arquivo binário deve iniciar com um cabeçalho de 72bits, sendo 56 bits formados pelos caracteres BARALHO e os 16 bits seguintes sendo um valor unsigned short com a quantidade de cartas armazenadas no arquivo. Na hora de ler o arquivo, o programa deve verificar se o arquivo contém a palavra BARALHO nos 56 bits iniciais, e usar o valor da quantidade de cartas armazenadas para simplificar a leitura do arquivo.
6. O jogo deve inicialmente selecionar 4 cartas aleatórias do baralho (as cartas sorteadas podem ser iguais) para cada jogador e armazená-las em dois vetores estáticos, um para cada jogador. Os vetores não devem guardar diretamente os dados das cartas escolhidas, eles **devem guardar apenas ponteiros**, que devem apontar para as cartas no vetor baralho. Pesquise como gerar números pseudoaleatórios em C++.
7. O Gerador e o Jogo devem ser programas distintos, criados em projetos separados dentro da mesma solução.
8. O Gerador e o Jogo devem compartilhar o mesmo arquivo binário. O Gerador cria e atualiza o baralho, enquanto o jogo utiliza as cartas armazenadas no baralho. Ambos os programas devem procurar o arquivo binário na pasta da solução, mas sem usar um caminho completo predefinido. Use a notação “. . \” para fazer referência à pasta anterior a pasta atual.
9. O Gerador pode inserir cartas via opção cadastrar ou via importação dos dados de um arquivo texto. **Forneça um arquivo texto de testes, com pelo menos oito cartas**, junto com o código fonte dos dois projetos.

ENTREGA DO TRABALHO

Grupos: Trabalho individual

Data da entrega: 19/03/2019 (até a meia noite)

Valor do Trabalho: 3,0 pontos (na 3a Unidade)

Forma de entrega: enviar apenas os arquivos fonte (.cpp), os arquivos de inclusão (.h) e um arquivo de testes da importação (.txt), compactados no formato **zip** através da tarefa correspondente no SIGAA.

O não cumprimento das orientações resultará em **penalidades:**

- Programa não executa no Visual Studio 2017 (3,0 pontos)
- Programa contém partes de outros trabalhos (3,0 pontos)
- Atraso na entrega (1,5 pontos por dia de atraso)
- Arquivo compactado em outro formato que não zip (0,5 ponto)
- Envio de outros arquivos que não sejam os solicitados (0,5 ponto)
- Programa sem comentários e/ou desorganizado (0,5 ponto)