

C++ Multiplataforma e Orientação a Objetos

Exercícios da parte 5:

Desenvolvimento web/CGI com C++ e
VBMcgi

Por: Sergio Barbosa Villas-Boas (sbVB)



<http://www.sbVB.com.br/c++.html>

Versão 8.0, de 06 de Outubro de 2003

Copyright © 1992~2003 by sbVB

Email do autor: sbvb@sbvb.com.br

Exercício 1)

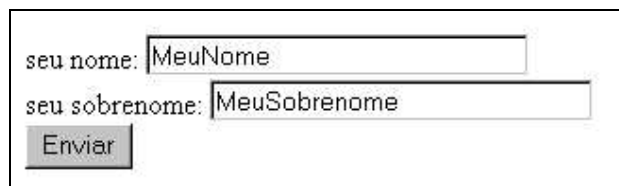
Faça um programa CGI que mostre a tabuada de um número inteiro, por exemplo $\text{int } n=4$; O browser deve mostrar “ $1 \times 4 = 4$
 $2 \times 4 = 8$ ” etc. Ou seja, uma linha debaixo da outra (o tag `
` faz mudar de linha).

Exercício 2)

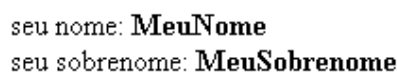
Seja a página web com formulário, como mostrado abaixo.

```
<!-- mywebpage.html -->
<form method="get" action="yourcgi.cgi">
seu nome: <input size="30" maxlength="30" name="$nome"><br>
seu sobrenome: <input size="30" maxlength="25" name="$sobrenome"><br>
<input type="submit" value="Enviar">
</form>
```

A figura abaixo ilustra o aspecto da página web relativo a esse formulário.



Faça um programa CGI, usando C++ e VBMcgi que mostre com negrito (bold) os conteúdos dos campos do formulário abaixo. A resposta do cgi no browser será como na figura abaixo. Use isolamento de camadas, isso é, deve haver um arquivo html separado que vai receber como parâmetro de troca o nome e o sobrenome.



Exercício 3)

Faça um programa CGI em c++ que tenha a seguinte funcionalidade. O programa deve mostrar no browser a data e hora do sistema, a partir do arquivo output.html, como mostrado abaixo. Obtenha a data e hora do sistema, crie strings com seus valores, e use VBMcgi para isolar-se do arquivo html (na camada de apresentação).


```
<!-- output.html -->
<html><body>
Hora do servidor:s_hour
Data do servidor: s_date
<!-- more data -->
</body></html>
```

Exercício 4)

Faça um programa CGI, com C++, que grave os dados de um formulário num arquivo de texto. Os dados devem ser acrescentados no final do arquivo, de modo que seja possível ler o arquivo e ver os dados de todos os usuários que preencheram o formulário.

```
<form method="POST" action="cgi-bin/yourcgi.cgi">
  Nome: <input type="text" name="nome" size="20"><br>
  Telefone: <input type="text" name="tel" size="20"><br>
  Endereço: <input type="text" name="end" size="20"><br>
  Email: <input type="text" name="email" size="20"><br>
  <input type="checkbox" name="bool_email" value="ON">
    Desejo receber email de propaganda<br>
  <input type="submit" value="Submit" name="B1">
</form>
```

O aspecto do formulário no browser é mostrado abaixo



Nome:

Telefone:

Endereço:

Email:

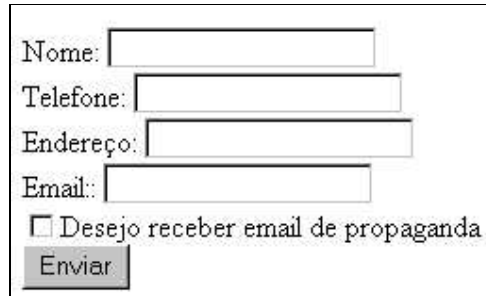
Desejo receber email de propaganda

Exercício 5)

Faça um programa CGI, com C++, que grave os dados de um formulário num arquivo de texto. Os dados devem ser acrescentados no final do arquivo, de modo que seja possível ler o arquivo e ver os dados de todos os usuários que preencheram o formulário.

```
<form method="POST" action="cgi-bin/yourcgi.cgi">
  Nome: <input type="text" name="nome" size="20"><br>
  Telefone: <input type="text" name="tel" size="20"><br>
  Endereço: <input type="text" name="end" size="20"><br>
  Email: <input type="text" name="email" size="20"><br>
  <input type="checkbox" name="bool_email" value="ON">
    Desejo receber email de propaganda<br>
  <input type="submit" value="Submit" name="B1">
</form>
```

O aspecto do formulário no browser é mostrado abaixo



Nome:
Telefone:
Endereço:
Email:
 Desejo receber email de propaganda

Exercício 6)

Faça um programa cgi que exiba uma página com o texto html:
“A página foi acessada x vezes”.

Sendo que x é o número de vezes que a página foi acessada.

Dica: o número x deve ser armazenado num arquivo no servidor.

Exercício 7)

Complete o programa CGI abaixo, para que tenha a seguinte funcionalidade: Seja o arquivo “sentences.txt”, que contém $n=10$ frases (uma por linha). O programa CGI deverá mostrar uma das dessas frases ao ser executado. Da próxima vez que for executado, deverá mostrar a próxima frase do arquivo. Assim sucessivamente, sendo que depois da exibição da última frase, retorna-se a primeira, num processo sem fim. O arquivo sentenceCount.dat armazena a informação de qual é a frase a ser exibida.

```
#include <fstream>
#include "vbmcgi.h"
int main()
{
    const int n = 10;
    const char *fraseFileName = "sentences.txt";
    const char *countFileName = "sentenceCount.dat";
    VBMcgi cgi;
    VBString sentence;
    ifstream readCountFile(countFileName);
    int count;
    readCountFile >> count;
    count++; // increment frase counter
    if (count > n)
        count = 1;
    readCountFile.close();
    // save in file counter value, for next time
    ofstream writeCountFile(countFileName);
    writeCountFile << count;
    writeCountFile.close();

    // add code here

    cgi.addBySource("s_sentence",sentence);
    cgi.out("sentence.html");
}
```

```
    return 0;
}
```

O arquivo `sentence.html` é mostrado abaixo.

```
<!-- sentence.html -->
<html><body>
A frase é: s_sentence
</body></html>
```

Como se poderia alterar esse programa para que se pudesse alterar facilmente a quantidade de frases em `sentences.txt`, sem necessidade de se recompilar o programa?

Exercício 8)

Complete o programa CGI abaixo para que liste os dados de um arquivo de dados "data.txt". Cada linha contém o nome, a nota 1 e nota 2 de um aluno, no formato "nome;nota1;nota2". Os dados devem ser listados por ordem alfabética, ou ordem de nota 1, ou ordem de nota 2, ou por ordem de nota média, conforme o botão pressionado no formulário html. O formato de dados é mostrado abaixo. O número de linhas do arquivo não é conhecida a priori.

```
nome1;nota1_1;nota1_2
nome2;nota2_1;nota2_2
nome3;nota3_1;nota3_2
...
nome_n;altura_n;peso_n
```

O formulário html é mostrado abaixo.

```
<form name="form1" method="post" action="yourcode.cgi">
  <input type="submit" name="byname" value="Ordenado por Nome">
  <input type="submit" name="bygrade1" value="Ordenado por Nota 1">
  <input type="submit" name="bygrade2" value="Ordenado por Nota 2">
  <input type="submit" name="byaverage" value="Ordenado por Média de notas">
</form>
```

O programa cgi parcial é mostrado abaixo. Preencha com código nos locais indicados.

```
// yourcode.cpp
#include "vbmcgi.h"
// if needed, insert code here (A)
int main()
{
    VBMcgi cgi;
    cgi.formDecode();
    VBString s = cgi.getVarContent("submit");
    const char *fileName = "data.txt";
    // if needed, insert code here (B)
    if (s=="byname")
    {
        // insert here code for order by name (C)
    }
    if (s=="bygrade1")
    {
        // insert here code for order by grade 1 (D)
    }
    if (s=="bygrade2")
```

```
{
  // insert here code for order by grade 2 (E)
}
if (s=="byaverage")
{
  // insert here code for order by average (F)
}
return 0;
}
```

Exercício 9)

Faça uma função “shortName”, que receba como entrada uma string, que seria um nome de pessoa. O retorno da função é o nome abreviado segundo as regras abaixo:

- 1) Um nome é abreviado por usar a primeira letra seguida de ponto. Exemplo: “Barbosa” => “B.”
- 2) O primeiro e último nome não se abreviam (fica com na entrada). Os demais são abreviados Exemplo: “Mário Augusto Nunes Gonçalves” => “Mário A. N. Gonçalves”.
- 3) As expressões “de, da, dos” devem ser suprimidas e não abreviadas. Exemplo: “João Batista da Silva” => “João B. Silva”. Observação: deve ser fácil acrescentar novas expressões a serem suprimidas.

Faça um programa CGI que leia um nome de pessoa e mostre no browser o nome abreviado. Mostre o programa fonte em C++ do programa CGI e os códigos html utilizados.