Pontifícia Universidade Católica do Rio Grande do Sul
Faculdade de Informática - FACIN

LAPRO II

Prof. Dr. Rafael Garibotti

- Baseado na prova aplicada em 2016/1 pelo Prof. **Dr. Edson Moreno**

AULA SOBRE:

REVISÃO PARA P1

REVISÃO PARA P1

Questão 1. *(2,5 pontos)* – Variáveis lógicas tradicionais podem assumir dois valores lógicos: verdadeiro ou falso. Implemente uma classe TRISTATE para objetos lógicos que podem assumir três valores correspondentes a verdadeiro, falso ou indeterminado.

- ✓ A classe deverá conter os métodos *set()*, *reset()* e *dontknow()* que colocam o objeto nos estados V, F e I, respectivamente;

- ✓ Sempre que uma variável é criada ela é colocada no estado I. Inclua também um método *get()* que retorna um caracter 'V', 'F' ou 'I', representando o valor do objeto.

REVISÃO PARA P1

Questão 1.

✓ **tristate.h**

```
#include <iostream>

using namespace std;

class tristate{
  private:
    char state;
  public:
    tristate();
    void set();
    void reset();
    void dontknow();
    char get();
};
```

✓ **tristate.cpp**

```
#include "tristate.h"
tristate::tristate(){
   state='I';
}
void tristate::set(){
   state='V';
}
void tristate::reset(){
   state='F';
}
void tristate::dontknow(){
   state='I';
}
char tristate::get(){
   return state;
}
```

✓ **main.cpp**

```
#include "tristate.h"

int main() {
   tristate val;
   cout << val.get() << endl;
   val.set();
   cout << val.get() << endl;
   val.reset();
   cout << val.get() << endl;
   val.dontknow();
   cout << val.get() << endl;
   return 0;
}
```

REVISÃO PARA P1

Questão 2. *(2,5 pontos)* – Implemente uma classe que modele um forno à gas. O botijão de gás do forno armazena no máximo 50 kg de gás. O forno consome 0,1 kg/hora.

- ✓ Ao ser criado, o forno tem seu botijão vazio. *(0,3 ponto)*;
- ✓ A partir da classe deve ser possível:
  - Abastecer o botijão com uma certa quantidade de gás *(0,2 ponto)*;
  - Acender o forno por um determinado número de horas. O forno apaga-se automaticamente após este período *(0,3 ponto)*;
  - Obter a quantidade de gás e o tempo de uso do forno desde que o botijão foi abastecido *(0,5 ponto)*;
- ✓ No programa principal,
  - Crie 2 fornos *(0,2 ponto)*;
  - Abasteça 10 kg no primeiro e 20 kg no segundo *(0,3 ponto)*;
  - Acenda o primeiro por 2 horas e meia e o segundo por 45 minutos *(0,3 ponto)*;
  - Exiba na tela o total de gás restante para cada botijão e quanto tempo cada forno esteve aceso desde que foi abastecido *(0,4 ponto).*

REVISÃO PARA P1

Questão 2.

✓ **forno.h**

```
#include <iostream>
using namespace std;
class forno{
  private:
    float botijao;
    float uso;
  public:
    forno();
    void abastecer(float
_valor);
    void acender(float
_horas);
    float tempoUso();
    float gasRestante();
};
```

✓ **forno.cpp**

```
#include "forno.h"
forno::forno(){uso = 0; botijao = 0;}
float forno::tempoUso(){return uso;}
float forno::gasRestante(){return botijao;}
void forno::abastecer(float _valor){
  this->uso = 0;
  this->botijao += _valor;
  if(this->botijao > 50)
    this->botijao = 50;
}
void forno::acender(float _horas){
  float consumo = (0.1*_horas);
  if(consumo <= botijao){
    botijao -= consumo;
    uso += _horas;
  }
  else{
    uso += (botijao/0.1);
    botijao = 0;
  }
}
```

✓ **main.cpp**

```
#include "forno.h"
int main() {
  forno f1, f2;
  f1.abastecer(10);
  f2.abastecer(20);
  f1.acender(2.5);
  f2.acender(0.75);
  cout << f1.tempoUso() << ", " <<
f1.gasRestante() << endl;
  cout << f2.tempoUso() << ", " <<
f2.gasRestante() << endl;
  return 0;
}
```

REVISÃO PARA P1

Questão 3. *(2,5 pontos)* – Crie um programa que leia um arquivo texto ENTRADA.TXT, contendo linhas de texto. Cada linha é formada por palavras separadas por um espaço em branco. Cada palavra pode ter de 1 a 10 caracteres. Como saída o programa deve gerar um segundo arquivo texto (SAIDA.TXT), com 10 linhas, no qual cada linha "i" contenha a quantidade de palavras de ENTRADA.TXT com tamanho "i". A seguir é apresentado um exemplo de um arquivo SAIDA.TXT gerado pela execução do programa sobre o arquivo ENTRADA.TXT.

- ✓ *Para calcular o tamanho de um objeto da classe string, você pode usar o método* ***int string::size()****.*
- ✓ *Para calcular o tamanho de uma string de C, você pode usar a função* ***int strlen(char *)***

Arquivo ENTRADA.TXT

Linha 1	Como a aurora precursora
Linha 2	do farol da divindade
Linha 3	Foi o vinte de setembro
Linha 4	O precursor da liberdade

Arquivo SAIDA.TXT

Linha 1	3
Linha 2	4
Linha 3	1
Linha 4	1
Linha 5	2
Linha 6	1
Linha 7	0
Linha 8	1
Linha 9	3
Linha 10	1

REVISÃO PARA P1

Questão 3. → Versão: *int string::size()*

✓ **main.cpp**

```
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main() {
  string linha;
  ifstream arqEntrada;
  ofstream arqSaida;
  int contador[10], aux;

  for(int i=0;i<10;i++)
    contador[i]=0;

  arqEntrada.open("entrada.txt");
  arqSaida.open("saida.txt");
  ...
  do{
    getline(arqEntrada, linha);
    aux=0;
    for(int i=0;i<linha.size();i++){
      if(linha[i]==' '){
        if(aux!=0){
          contador[aux-1]+=1;
          aux=0;
        }
      }
      else
        aux++;
    }
    if(aux)
      contador[aux-1]+=1;
  }while(!arqEntrada.eof());

  arqSaida << contador[0];
  for(int i=1; i<10; i++)
    arqSaida << endl << contador[i];

  arqEntrada.close();
  arqSaida.close();
  return 0;
}
```

REVISÃO PARA P1

Questão 3. → Versão: *int strlen(char *)*

✓ **main.cpp**

```
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main() {
  char palavra [50];
  ifstream arqEntrada;
  ofstream arqSaida;
  int contador[10], aux;

  for(int i=0;i<10;i++)
    contador[i]=0;

  arqEntrada.open("entrada.txt");
  arqSaida.open("saida.txt");
  ...
  do{
    arqEntrada >> palavra;
    aux = strlen(palavra);
    contador[aux-1] += 1;
  }while(!arqEntrada.eof());

  arqSaida << contador[0];
  for(int i=1; i<10; i++)
    arqSaida << endl << contador[i];

  arqEntrada.close();
  arqSaida.close();
  return 0;
}
```

REVISÃO PARA P1

Questão 4. *(2,5 pontos)* – Crie os cabeçalhos dos métodos que sobrecarreguem os operadores de soma, subtração e multiplicação para objetos da classe "*ponto*" descrito abaixo. Os métodos devem ser internos à classe "*ponto*".

Sobrecargas a serem criadas:

- ✓ *Um operador de soma capaz de somar as coordenadas de dois pontos.*
- ✓ *Um operador de soma capaz de somar um ponto e um número real.*
- ✓ *Um operador binário de subtração capaz de subtrair as coordenadas de dois pontos.*
- ✓ *Um operador unário de subtração capaz trocar o sinal das coordenadas do operando.*
- ✓ *Um operador de multiplicação capaz de multiplicar as coordenadas de dois pontos.*

```
class ponto{
  private:
    float x,y;
  public:
    ponto(float _x=0, float _y=0);
    float getX();
    float getY();
    void setX(float _x);
    void setY(float _y);
};
```

REVISÃO PARA P1

Questão 4.

✓ **ponto.h**

```
#include <iostream>

using namespace std;

class ponto{
  private:
    float x,y;
  public:
    ponto(float _x = 0, float
_y = 0);
    float getX();
    float getY();
    void setX(float _x);
    void setY(float _y);
    ponto operator+(ponto _b);
    ponto operator+(float _b);
    ponto operator-(ponto _b);
    ponto operator-();
    ponto operator*(ponto _b);
};
```

✓ **ponto.cpp**

```
#include "ponto.h"

ponto::ponto(float _x, float _y){x
= _x; y = _y;}
float ponto::getX(){return x;}
float ponto::getY(){return y;}
void ponto::setX(float _x){x = _x;}
void ponto::setY(float _y){y = _y;}

ponto ponto::operator+ (ponto _b){
  ponto tmp;
  tmp.setX(this->x+_b.getX());
  tmp.setY(this->y+_b.getY());
  return tmp;
}

ponto ponto::operator+ (float _b){
  ponto tmp;
  tmp.setX(this->x+_b);
  tmp.setY(this->y+_b);
  return tmp;
}

ponto ponto::operator- (ponto _b){
  ponto tmp;
  tmp.setX(this->x-_b.getX());
  tmp.setY(this->y-_b.getY());
  return tmp;
}
...
```

✓ **ponto.cpp**

```
...
ponto ponto::operator- (){
  ponto tmp;
  tmp.setX(-this->x);
  tmp.setY(-this->y);
  return tmp;
}

ponto ponto::operator* (ponto _b){
  ponto tmp;
  tmp.setX(this->x*_b.getX());
  tmp.setY(this->y*_b.getY());
  return tmp;
}
```

✓ **main.cpp**

```
#include "ponto.h"
int main() {
  ponto a(1, 1), b(2, 2), c;
  c=a+b;
  cout << c.getX() << "," << c.getY() << endl;
  c=a+5.5;
  cout << c.getX() << "," << c.getY() << endl;
  c=a-b;
  cout << c.getX() << "," << c.getY() << endl;
  c=-b;
  cout << c.getX() << "," << c.getY() << endl;
  c=(a+b)*b;
  cout << c.getX() << "," << c.getY() << endl;
  return 0;
}
```