

```
//conteúdo do arquivo pacote.h
#ifndef PACOTE_H
#define PACOTE_H
    #include <string>
    #include <iostream>
    class Pacote
    {
        friend istream &operator>> (istream &, Pacote &);
        friend ostream &operator<< (ostream &, Pacote &);
    public:
        Pacote(double = 1, double = getCustoPorQuilo());
        static void setCustoPorQuilo(double);
        static double getCustoPorQuilo();
        virtual double calculaCusto();
        ...
    };
#endif
```

```
//conteúdo do arquivo principalpacote.cpp
#include "pacote.h"
#include "pacotedoisdias.h"
#include "pacotenoite.h"
#include <iostream>
using namespace std;
#define MAX 10
main()
{
    int opcao, numeroDeElementos=0;
    Pacote *vetor[MAX];
    do
    {
        cout << endl << "Digite:" << endl <<
        "1 - Inserir um pacote para entrega em dois dias;" <<
        endl << "2 - Inserir um pacote para entrega a noite;" << endl <<
        "3 - Imprimir enderecos para postagem e custos da postagem;" <<
        endl << "4 – Imprimir custos total das postagens." << endl <<
        "5 - Finaliza o programa." << endl << "Opcao? ";
        cin >> opcao;
```

```

switch(opcao)
{
  case 1:
    if (numeroDeElementos<MAX)
    {
      vetor[numeroDeElementos] = new PacoteDoisDias;
      cin >> (*vetor[numeroDeElementos]);
      numeroDeElementos++;
    }
    else
      cout << endl << "Nao ha espaco para mais pacotes" << endl;
      break;
  case 2:
    if (numeroDeElementos<MAX)
    {
      vetor[numeroDeElementos]= new PacoteNoite;
      cin >> (*vetor[numeroDeElementos]);
      numeroDeElementos++;
    }
    else
      cout << endl << "Nao ha espaco para mais pacotes" << endl;
      break;
}

```

case 3:

```
for (int i=0; i<numeroDeElementos; i++)  
    cout << (*vetor[i]) << endl << endl << endl <<  
    "Custo da postagem: " << vetor[i]->calculaCusto();  
break;
```

case 4:

```
{  
    double auxiliar=0;  
    for (int i=0; i<numeroDeElementos; i++)  
        auxiliar += vetor[i]->calculaCusto();  
    cout << endl << "Custo total das postagens: " <<  
    auxiliar << endl;  
}  
break;
```

case 5:

```
cout << endl << "Obrigado por utilizar nosso software." << endl;  
break;
```

default:

```
cout << endl << "Opcao invalida!" << endl;
```

```
}
```

```
}while(opcao!=5);
```

```
return 0;
```

Linguagem de Programação C++

Polimorfismo

Pode-se especificar uma classe abstrata em C++. Para tal, defini-se uma função-membro virtual como uma função virtual pura. Desta forma, a classe base que contém uma função virtual pura não pode ser instanciada.

Uma função virtual é definida como pura igualando seu protótipo a zero. Vejamos um exemplo:

```

#include <iostream>
using namespace std;
class Base {
    public:
        virtual void func()=0;
};
class Derivada: public Base {
    public:
        void func()
        {
            cout << "Esta eh func() de derivada" << endl;
        }
};
int main()
{
    Base *ptr;
    Derivada objeto1;
    ptr = &objeto1;
    ptr->func();
    Base objeto2; //Erro: pois a classe base é abstrata
    return 0;
}
425 }

```

Linguagem de Programação C++

Polimorfismo

Um aluno atento deve ter se perguntado:

Existe uma maneira de determinar em tempo de execução para qual das classes derivadas um ponteiro para a classe base está apontando?

A resposta é sim. A linguagem C++ disponibiliza para esta finalidade o *dynamic_cast*.

O exemplo a seguir o utiliza.

```

#include <iostream>
using namespace std;
class Base {
    public:
        virtual void func()=0; };
class Derivada1: public Base {
    public:
        void func() { cout << "Esta eh func() de derivada1" << endl; } };
class Derivada2: public Base {
    public:
        void func() { cout << "Esta eh func() de derivada2" << endl; } };
int main()
{
    Base *ptr;
    Derivada1 objeto1, *ptr2;
    Derivada2 objeto2;
    ptr = &objeto1;
    ptr2 = dynamic_cast <Derivada1 *> (ptr);
    if (ptr2)
        ptr->func();
    ptr = &objeto2;
    ptr->func();
    return 0;
}

```


Linguagem de Programação C++

Exercício:

Utilizando este conceito implemente uma nova opção no menu do exercício do slide 419 a qual possibilita determinar o custo total apenas dos pacotes que serão entregues à noite.