

```

1  /* ----- */
2  /*  Aluno    78865  */
3  /*  Miquel   Amaral  */
4  /*  Quarta Feira 15h30  */
5  /* ----- */
6
7
8  /* ----- */
9  /*  PaymentMethod.cpp  */
10 /* ----- */
11 #include <string>
12
13 class PaymentMethod {
14 private:
15     std::string name;
16 public:
17     PaymentMethod(std::string name) : name(name) {}
18     std::string getName() { return name; }
19     void setName(std::string name) { name = name; }
20 };
21
22 /* ----- */
23 /*  ProductType.cpp  */
24 /* ----- */
25 #include <string>
26
27 class ProductType {
28 private:
29     std::string name;
30 public:
31     ProductType(std::string name) : name(name) {}
32     std::string getName() { return name; }
33     void setName(std::string name) { name = name; }
34 };
35
36 /* ----- */
37 /*  Purchase.cpp  */
38 /* ----- */
39 #include <forward list>
40 #include <ctime>
41 #include "ProductType.cpp"
42 #include "PaymentMethod.cpp"
43
44 class Purchase {
45 private:
46     std::time_t date;
47     int value = 0;
48     std::shared_ptr<PaymentMethod> paymentMethod;
49     std::shared_ptr<ProductType> prodType;
50 public:
51     Purchase(std::time_t date, std::string payment, int value,
52             std::string prodType) : date(date), value(value){
53         paymentMethod = std::make_shared<PaymentMethod>(prodType);
54         prodType = std::make_shared<ProductType>(prodType);
55     }
56     int getValue(){
57         return value;
58     }
59     std::time_t getDate(){
60         return date;
61     }
62     void applyDiscount(float discount){
63         value = value * (1-discount);
64     }
65 };
66
67

```

```

68  /* ----- */
69  /*      Filter.hpp      */
70  /* ----- */
71  class Filter {
72  public:
73      virtual std::list< std::shared_ptr<Client>>
74          getFilteredList(Store *store) =0;
75  };
76
77  /* ----- */
78  /*      Filter.cpp      */
79  /* ----- */
80  #include <forward list>
81  #include <list>
82  //#include "Client.cpp"
83
84  class XPurchases : public Filter {
85  private:
86      int purchaseNumber;
87  public:
88      XPurchases(int purchaseNumber){
89          purchaseNumber = purchaseNumber;
90      }
91
92
93      std::list< std::shared_ptr<Client>> getFilteredList(Store *store){
94
95          std::cout << "A filtrar PURCHASES " << std::endl;
96          std::list< std::shared_ptr<Client>> returnList;
97          for (int i = 0; i < store->getSize(); i++) {
98              std::shared_ptr<Client> c = store->getClient(i);
99              if(c->getPurchaseListSize() >= purchaseNumber)
100                  returnList.push_back(std::move(c));
101          }
102          return returnList;
103      }
104
105  };
106
107  class XPoints : public Filter {
108  private:
109      int pointsMin;
110  public:
111      XPoints(int pointsMin){
112          pointsMin = pointsMin;
113      }
114
115
116      std::list< std::shared_ptr<Client>> getFilteredList(Store *store){
117          std::cout << "A filtrar POINTS " << std::endl;
118          std::list< std::shared_ptr<Client>> returnList;
119          for (int i = 0; i < store->getSize(); i++) {
120              std::shared_ptr<Client> c = store->getClient(i);
121              if(c->getPoints() >= pointsMin)
122                  returnList.push_back(std::move(c));
123          }
124          return returnList;
125      }
126
127  };
128
129  /* ----- */
130  /*      Client.cpp      */
131  /* ----- */
132  #include <forward list>
133  #include <list>
134  #include <ctime>

```

```

135 #include <string>
136 #include "Purchase.cpp"
137
138 class Client {
139 private:
140     std::string name;
141     long int contact;
142     std::time_t birthday;
143     std::time_t firstBuy;
144     std::list< std::shared_ptr<Purchase> > purchases;
145     int points;
146     float discount = 0;
147 public:
148     Client(std::time_t birthday, long int contact, std::string name) :
149         birthday(birthday), contact(contact), name(name){
150         points = 0;
151     }
152     void addPurchase(std::shared_ptr<Purchase> p){
153         points = points + p->getValue();
154         if( firstBuy){
155             firstBuy = p->getDate();
156         }
157         purchases.push_back(std::move(p));
158     }
159
160     void setDiscount(double discount){
161         discount = discount;
162     }
163
164     int getPurchaseListSize(){
165         return purchases.size();
166     }
167
168     int getPoints(){return points;}
169     void setPoints(int points){ points = points;}
170
171     std::string getName(){return name;}
172
173 };
174
175 /* ----- */
176 /*      Store.cpp      */
177 /* ----- */
178
179 #include <string>
180 #include "Client.cpp"
181 #include "Filter.hpp"
182
183 class Store {
184 private:
185     std::string name;
186     std::list< std::shared_ptr<Client> > clients;
187 public:
188     Store(std::string name) : name(name){}
189     void addClient(std::shared_ptr<Client> c){
190         clients.push_back(std::move(c));
191     }
192
193     std::list< std::shared_ptr<Client> > getList(std::shared_ptr<Filter> filter){
194         return filter->getFilteredList(this);
195     }
196
197     std::shared_ptr<Client> getClient(int index) {
198         std::list<std::shared_ptr<Client>>::iterator i = clients.begin();
199         std::advance(i, index);
200         return *i;
201     }

```

```

202
203 int getSize() {
204     return clients.size();
205 }
206 };
207
208 /* ----- */
209 /*      App.cpp      */
210 /* ----- */
211 #include <forward list>
212 #include <iostream>
213
214 class Store;
215
216 #include "Store.cpp"
217 #include "Filter.cpp"
218
219 int main(){
220     std::shared_ptr<Store> store = std::make_shared<Store>("loja de ferragens");
221     time_t currTime;
222     time_t tenYears;
223     time(&currTime);
224     tenYears = currTime - (10*365*24*60*60);
225
226
227     std::shared_ptr<Client> c1=std::make_shared<Client>(currTime,1111111,"Joao");
228     std::shared_ptr<Client> c2=std::make_shared<Client>(tenYears,9999999,"Jota");
229     store->addClient(c1);
230     store->addClient(c2);
231
232     std::shared_ptr<Purchase> p1 =
233         std::make_shared<Purchase>(currTime, "cash", 30, "milk");
234     std::shared_ptr<Purchase> p2 =
235         std::make_shared<Purchase>(currTime, "cash", 30, "eggs");
236     std::shared_ptr<Purchase> p3 =
237         std::make_shared<Purchase>(currTime, "cash", 30, "milk");
238     std::shared_ptr<Purchase> p4 =
239         std::make_shared<Purchase>(currTime, "credit card", 1000, "laptop");
240
241
242     c1->addPurchase(p1);
243     c1->addPurchase(p2);
244     c1->addPurchase(p3);
245
246     c2->addPurchase(p4);
247
248
249     std::shared_ptr<Filter> f2 = std::make_shared<XPurchases>(2);
250     std::list< std::shared_ptr<Client> > listaCompras2 = store->getList(f2);
251
252     std::list<std::shared_ptr<Client>>::iterator i;
253     for (i = listaCompras2.begin(); i != listaCompras2.end(); ++i) {
254         std::shared_ptr<Client> cliente = *i;
255         std::cout << cliente->getName() << std::endl;
256     }
257
258
259     std::shared_ptr<Filter> f3 = std::make_shared<XPoints>(100);
260     std::list< std::shared_ptr<Client> > listaPontos400 = store->getList(f3);
261     for (i = listaPontos400.begin(); i != listaPontos400.end(); ++i) {
262         std::shared_ptr<Client> cliente = *i;
263         cliente->setDiscount(0.50);
264         std::cout << cliente->getName() << std::endl;
265     }
266     return 0;
267 }
268

```