

```

140 /**
141  * @brief peek operation, peeking the latest promoted model at the top of the stack (without popping)
142  *      Both time and auxiliary space complexity need to be O(1)
143  * @param
144  * @return PromotedModel
145  */
146 PromotedModel PromotedCarModelStack::peek() {
147
148     //Checks to see if carNameAndPrice vector is empty
149     if (carNameAndPrice.empty()){
150         throw std::logic_error("Promoted car model stack is empty");
151     }
152
153     return carNameAndPrice.back();
154 }
155
156 /**
157  * @brief getHighestPricedPromotedModel,
158  *      getting the highest priced model among the past promoted models
159  *      Both time and auxiliary space complexity need to be O(1)
160  * @param
161  * @return maxModel
162  */
163 PromotedModel PromotedCarModelStack::getHighestPricedPromotedModel() {
164
165     //Checks to see if carNameAndPrice vector is empty
166     if (carNameAndPrice.empty()){
167         throw std::logic_error("Promoted car model stack is empty");
168     }
169
170     return maxModel;
171 }
172
173 /**
174  * @brief getLowestPricedPromotedModel,
175  *      getting the lowest priced model among the past promoted models
176  *      Both time and auxiliary space complexity need to be O(1)
177  * @param
178  * @return minModel
179  */
180 PromotedModel PromotedCarModelStack::getLowestPricedPromotedModel() {
181
182     //Checks to see if carNameAndPrice vector is empty
183     if (carNameAndPrice.empty()){
184         throw std::logic_error("Promoted car model stack is empty");
185     }
186
187     return minModel;
188 }

```