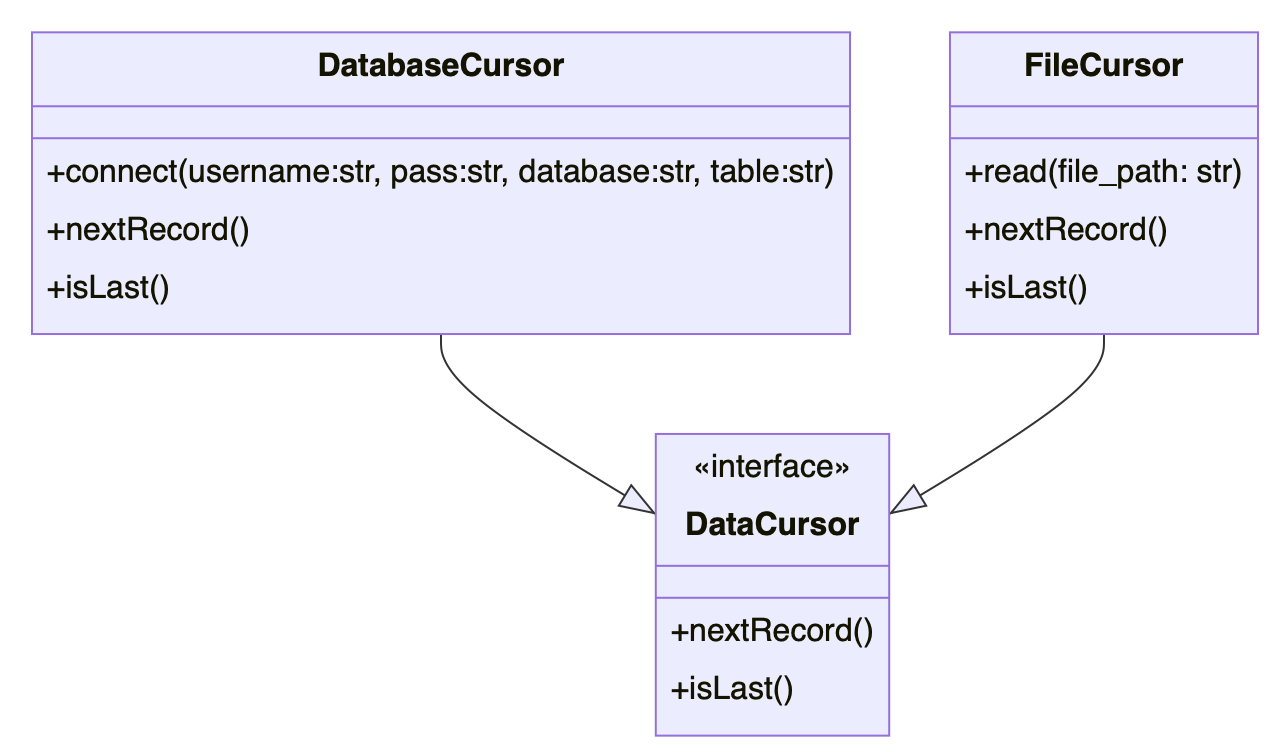
Vārds Uzvārds:  
  
Studenta numurs:

## 1. Uzdevums (33%)

Balstoties uz UML diagrammu izvēlēties pareizo programmēšanas šablona variantu.

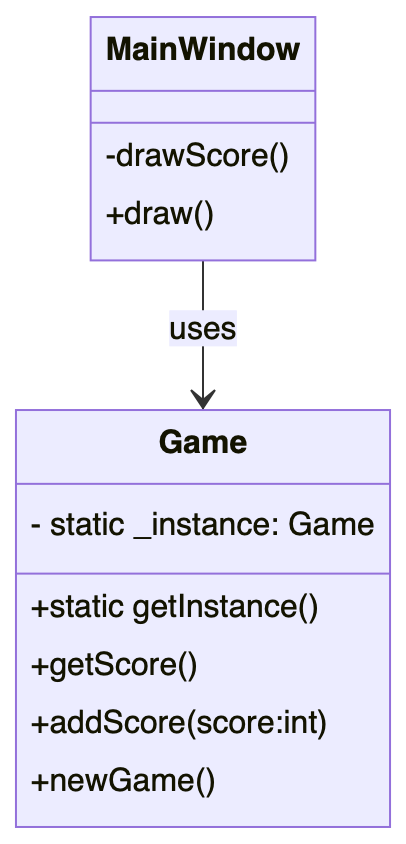
1.1. Uzdevums

Atbilžu varianti:  
a) Singleton  
b) Adapter  
c) Iterator



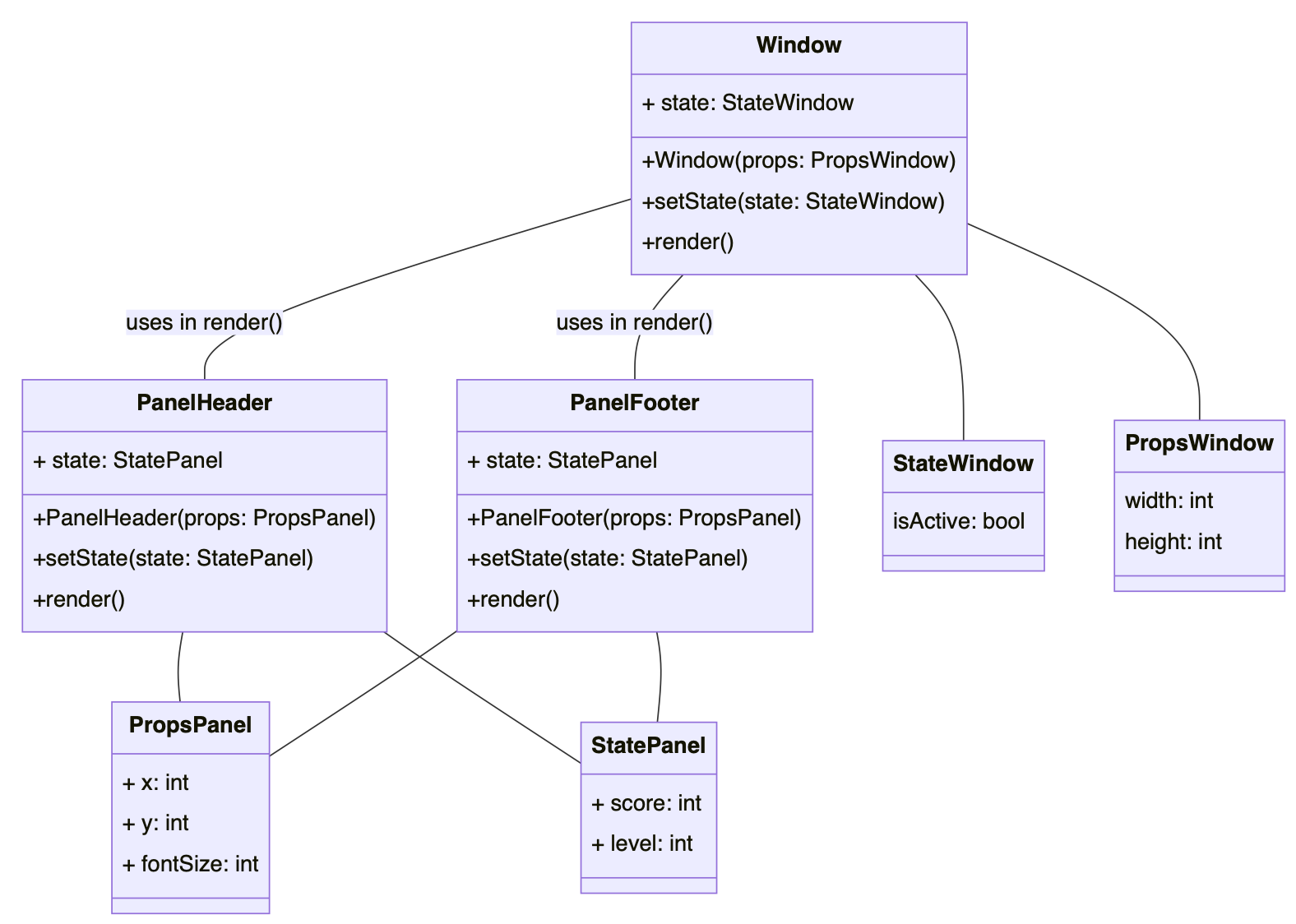
1.2. Uzdevums

Atbilžu varianti:  
a) Decorator  
b) Adapter  
c) Singleton



1.3. Uzdevums

Atbilžu varianti:  
a) Closure  
b) Adapter  
c) React



## 2. Uzdevums (33%)

Uzzīmēt UML diagrammu vienkāršotai tīmekļa iepirkšanās platformas versiju, izmantojot MVC (Model-View-Controller) modeli. Izveidojiet modeli Produktam (Product), Iepirkumu grozam (ShoppingCart) un Lietotājam (User). Produkta modelim jāiekļauj īpašības, piemēram, product\_id, name, description, price, amount. Iepirkumu grozam jāspēj pievienot vai noņemt Produktu objektus. Lietotājam īpašības, piemēram, user\_id, name, e\_mail un iepirkumu grozs. Jābūt iespējai izvadīt produktu sarakstu, viena produkta detaļas un lietotāja iepirkumu groza saturu. Implementējiet funkcijas, lai apstrādātu lietotāju darbības, piemēram, produktu pievienošanu/noņemšanu no iepirkumu groza un pirkšanas procesu.

## 3. Uzdevums (33%)

Dots programmatūras pirmkods Python valodā, lai ievadītu atslēgas vārdus, izmantojot programmatūras šablonu un dota funkcija atcelt (undo) un atkārtot (redo) pievienotos atslēgas vārdus.   
Nosaukt kas ir šajā programatūras kodā dotais šablona nosaukums un atzīmēt kļūdas programmatūras pirkmodā tās apvelkot vai nosvīrtot un pierakstīt pareizu programmatūras pirmkodu:

class Event:  
 def execute():  
 pass  
 def undo():  
 pass

class TypeEvent:  
 def \_\_init\_\_(text, keywords):  
 self.\_text = text  
 self.\_keywords = keywords  
 def execute():  
 self.\_keywords.append(self.\_text)  
 def undo():  
 self.\_keywords.pop()  
  
class History(Event):  
 def \_\_init\_\_():  
 self.\_history = []  
 self.\_index = -1  
 def execute(event):  
 self.\_history = self.\_history[:self.\_index]  
 self.\_history.append(event)  
 self.\_index += 1  
 event.execute()  
 def undo():  
 self.\_history[self.\_index].undo()  
 self.\_index -= 1  
 def redo():  
 self.\_index += 1  
 self.\_history[self.\_index].execute()  
  
history = History()  
while True:  
 text = input("Enter new keyword, 'undo', or 'redo': ")  
 keywords = []  
  
 if text == 'undo':  
 history.undo()  
 elif text == 'redo':  
 history.redo()  
 else:  
 event = TypeEvent(text, keywords)  
 history.execute(event)  
  
 print(keywords)

Programmatūras primoda darbības piemērs (pēc kļūdu salabošanas):

Enter new keyword, 'undo', or 'redo': >? aaa  
['aaa']  
Enter new keyword, 'undo', or 'redo': >? bbb  
['aaa', 'bbb']  
Enter new keyword, 'undo', or 'redo': >? ccc  
['aaa', 'bbb', 'ccc']  
Enter new keyword, 'undo', or 'redo': >? undo  
['aaa', 'bbb']  
Enter new keyword, 'undo', or 'redo': >? redo  
['aaa', 'bbb', 'ccc']  
Enter new keyword, 'undo', or 'redo': >? ddd  
['aaa', 'bbb', 'ccc', 'ddd']

Programmatūras šablona nosaukums (ierakstīt): [ ]