++rr

**МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики**

**Кафедра програмного забезпечення комп’ютерних систем**

**Лабораторна робота № 4**

**з дисципліни “Об’єктно орієнтоване програмування”**

**тема “Реалізація поведінкових шаблонів проектування”**

|  |  |  |
| --- | --- | --- |
| **Виконав**  **студент II курсу**  **групи КП-83**  **Коваль Андрій Олекснадрович**  **(прізвище, ім’я, по батькові)** |  | **Перевірив**  **“\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.**  **викладач**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **(прізвище, ім’я, по батькові)** |

**Київ 2020**

**Мета роботи**

Ознаойомлення з основними характеристиками шаблонів «Стратегія», «Шаблонний метод» та «Стан», запам’ятовування поширених ситуацій, коли використання цих шаблонів є доцільним, набуття вмінь та навичок реалізації шаблонів під час створення програмного коду.

**Постановка завдання**

Варіант 8

1) За допомогою шаблону проектування реалізувати автоматичний розрахунок знижок у сплаті за абонементи у спортивному комплексі. Якщо людина є клієнтом більше півроку, вона має знижку 5% (статус «Постійний клієнт»). Якщо людина оформлює місячний абонемент до спорткомплексу регулярно (рівно через місяць від дати попереднього придбання абонементу), вона має додатково 10% знижки (статус «VIP клієнт»).

2) В залежності від кількості часу, який домогосподарка може витратити на прибирання квартири, існує 3 варіанти прибирання: легке прибирання (скласти розкидані речі, підмести підлогу), звичайне прибирання (витерти пил, використати пилосос) та генеральне прибирання (вологе прибирання підлоги, а також миття вікон). За допомогою шаблону проектування реалізувати дані види прибирання.

**Задача 1**

**Обґрунтування вибору шаблону**

Шаблон «Стан» використовується для зміни поведінки об’єкта в залежності від стану. У нашому випадку стани слугують методами для зміни відсоткової ставки знижки відвідувача спортклубу.

**Діаграма класів**

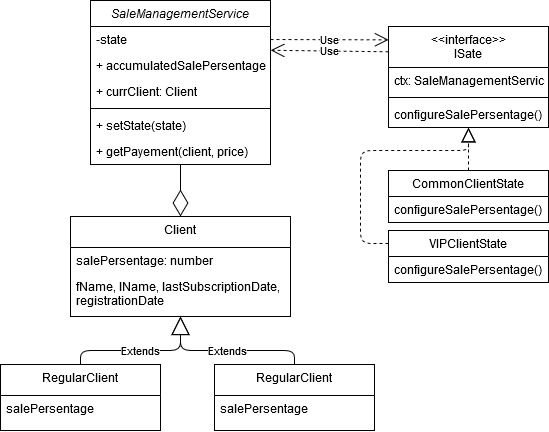


Рис. Діаграма класів завдання 1

**Фрагменти коду програми**

|  |
| --- |
| **Система керування знижками** |
| |  | | --- | |  |   **export** **class** SaleManagementService {  **private** state: **IState**;  **public** accumulatedSalePersentage: **number** = **0**;  **public** currClient: **Client**;  **constructor**() {  **this**.state = **new** CommonClientState(**this**);  }  **public** setState(state: **IState**) {  **this**.state = state;  **this**.state.configureSalePersentage();  }  **public** getPayement(client: **Client**, price: **number**) {  **this**.currClient = client;  **this**.state.configureSalePersentage();  **const** persentage = **this**.accumulatedSalePersentage;  **this**.accumulatedSalePersentage = **0**;  **this**.state = **new** CommonClientState(**this**);  **return** price \* (**100** - persentage) / **100**;  }  } |

|  |
| --- |
| **IState** |
| **export** **interface** IState {  ctx: **SaleManagementService**;  configureSalePersentage();  } |
| **Усі стани** |
| **export** **class** CommonClientState **implements** IState {  **constructor**(**public** ctx: **SaleManagementService**) { }  configureSalePersentage() {  **const** isRegularClient: **boolean** = dateDiffInDays(**new** Date(), **this**.ctx.currClient.registrationDate) >= **365** / **2**;  console.log('isRegularClient:', isRegularClient)  **const** isVIPClient: **boolean** = dateDiffInDays(**new** Date(), **this**.ctx.currClient.lastSubscriptionDate) <= **30**;  console.log('isVIPClient:', isVIPClient)  **if** (isRegularClient) {  **this**.ctx.setState(**new** RegularClietnState(**this**.ctx));  }  **if** (isVIPClient) {  **this**.ctx.setState(**new** VIPClientState(**this**.ctx));  }  }  }  **export** **class** RegularClietnState **implements** IState {  **constructor**(**public** ctx: **SaleManagementService**) { }  configureSalePersentage() {  **this**.ctx.accumulatedSalePersentage += RegularClient.salePersentage;  }  }  **export** **class** VIPClientState **implements** IState {  **constructor**(**public** ctx: **SaleManagementService**) { }  configureSalePersentage() {  **this**.ctx.accumulatedSalePersentage += VIPClient.salePersentage;  }  } |

|  |
| --- |
| **main.ts** |
| **const** salesService = **new** SaleManagementService();  **const** payment1 = salesService.getPayement(**new** Client('andrii', 'koval', **new** Date('03/08/2020'), **new** Date('2017/07/25')), **100**);  console.log('payment1:', payment1);  **const** payment2 = salesService.getPayement(**new** Client('andrii', 'koval', **new** Date('03/01/2020'), **new** Date('2020/07/25')), **111**);  console.log('payment2:', payment2); |

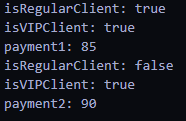
**Результати роботи програми**

Рис. Результат роботи завдання 1

**Задача 2**

**Обґрунтування вибору шаблону**

Шаблон «Стратегія» використовується виокремлення сімейств алгоритмів та дій. У нашому випадку буде змінюватися вид прибирання, тобто його «стратегія» відповідно до кількості відведеного часу.

**Діаграма класів**

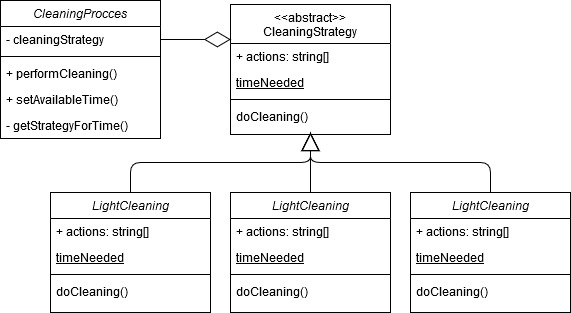


Рис. Діаграма класів завдання 2

**Фрагменти коду програми**

|  |
| --- |
| **Клас процесу прибирання** |
| **export** **class** CleaningProcces {  **private** cleaningStrategy: **CleaningStrategy**;  **constructor**(timeAvailable: **number**) {  **this**.cleaningStrategy = **this**.getStrategyForTime(timeAvailable);  }  **public** performCleaning() {  **this**.cleaningStrategy.doCleaning();  }  **public** setAvailableTime(time: **number**) {  **this**.cleaningStrategy = **this**.getStrategyForTime(time);  }  **private** getStrategyForTime(timeAvailable: **number**): CleaningStrategy {  **const** strategy = [CommonCleaning, SuperCleaning, LightCleaning]  .filter( st => st.timeNeeded <= timeAvailable )  .sort( (a, b) => a.timeNeeded - b.timeNeeded )  .pop() || **null**;  **if** (strategy == **null**)  **throw** **new** Error('Not enough time to perform cleaning');  **return** **new** strategy();  }  } |

|  |
| --- |
| **Абстрактна стратегія прибирання** |
| **export** **abstract** **class** CleaningStrategy {  actions: **string**[];  **static** readonly timeNeeded: **number**;  doCleaning() {  console.log(**this**.actions.join('\n'));  }  } |

|  |
| --- |
| **Різні стратегії прибирання** |
| **export** **class** LightCleaning **extends** CleaningStrategy {  actions: **string**[] = [  'Picking up thrown clothes',  'Sweeping the floor'  ];  **public** **static** readonly timeNeeded = **1**;  doCleaning() {  console.log('Doing light cleaning--');  **super**.doCleaning();  }  }  **export** **class** CommonCleaning **extends** CleaningStrategy {  actions: **string**[] = [  'Wiping off the dust',  'Vacuuming the floor'  ];  **public** **static** readonly timeNeeded = **3**;  doCleaning() {  console.log('Doing common cleaning');  **super**.doCleaning();  }  }  **export** **class** SuperCleaning **extends** CleaningStrategy {  actions: **string**[] = [  'Wet cleaning the floor',  'Cleaning the windows'  ];  **public** **static** readonly timeNeeded = **5**;  doCleaning() {  console.log('Doing superMegaXXLProMaxPlus cleaning');  **super**.doCleaning();  }  } |

|  |
| --- |
| **Клієнтський код** |
| **try** {  **const** cleaning = **new** CleaningProcces(timeAvailable);  cleaning.performCleaning();  } **catch** {  console.log('Unfortunatelly mon doest have a time for this(');  } |

**Результати роботи програми**

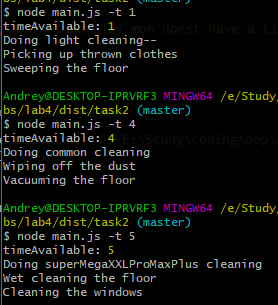


Рис. 3 приклади роботи з різною кількістю відведеного часу

**Висновки**

Я ознайомився з основними характеристиками шаблонів «Стратегія», «Шаблонний метод» та «Стан», запам’ятав поширені ситуації, коли використання цих шаблонів є доцільним, набув вмінь та навичок реалізації шаблонів під час створення програмного коду.