Қазақстан Республикасы Оқу-ағарту министрлігі

«Білім» кәсіби гуманитарлық-техникалық колледжі

****

**Практикалық жұмыс №1**

**Тақырыбы: Компоненттік әдіс және CASE операторы**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Жұмыстың орындау сапасы | Баға диапазоны | Орындаған % |
| 1 | Орындалған жоқ, сабақта себепсіз болмады. | 0 % |  |
| 2 | Жұмыстың орындалуы және студенттің белсенділігі | 0-50% |  |
| 3 | Жұмысты рәсімдеу | 0-20% |  |
| 4 | Анықтамалар мен техникалық әдістемелерді, пәннің оқу-әдістемелік кешенін, лекция конспектілерін қолдана білу. | 0-5% |  |
| 5 | Техникалық құралдарды пайдалана білу | 0-5% |  |
| 6 | Жұмысты қорғау | 0-20% |  |
|  | Қорытынды | 0-100% |  |

**Оқытушы: Нургисаева У.М.**

**Студент:Жексенбай Ольга**

**Мамандығы: 2BQ-23**

**Тараз 2025**

**Практикалық жұмыс: Компоненттік әдіс және CASE операторы**

**Мақсаты:** Компоненттік әдісті қолдана отырып, бағдарламалық жүйенің құрылымын жасау.CASE операторы арқылы шешім қабылдау механизмін бағдарламалау.CASE технологияларын пайдаланып, жүйелік жобалау жұмыстарын жүргізу.

**1. Сұрақтарға жауап беріңіз:**

1. Компоненттік әдістің негізгі артықшылықтары қандай?
2. CASE технологиялары қандай кезеңдерде қолданылады?
3. CASE операторының **if-else** құрылымынан айырмашылығы қандай?
4. CASE технологияларын пайдалану бағдарламашының жұмысын қалай жеңілдетеді?

**Тапсырма 1: Компоненттік әдісті қолдану**

Берілген талаптарға сай шағын бағдарламалық жүйенің құрылымын жасаңыз:

1. Жүйе **пайдаланушының тапсырысын өңдейтін** модульдерден тұруы керек.
2. Компоненттер келесі функцияларды орындауы тиіс:
   * **Пайдаланушы енгізуін қабылдау**
   * **Тапсырысты өңдеу**
   * **Тапсырысты растау және шығару**
3. Әр компонент жеке модуль ретінде қарастырылып, негізгі бағдарлама оларды біріктіруі керек.

**Мысал (Python):**

python

class OrderInput:

def get\_order(self):

return input("Тапсырысты енгізіңіз: ")

class OrderProcessor:

def process\_order(self, order):

return f"Тапсырыс өңделді: {order}"

class OrderOutput:

def display\_order(self, processed\_order):

print(processed\_order)

# Компоненттерді біріктіру

order\_input = OrderInput()

order\_processor = OrderProcessor()

order\_output = OrderOutput()

user\_order = order\_input.get\_order()

processed\_order = order\_processor.process\_order(user\_order)

order\_output.display\_order(processed\_order)

📌 **Тапсырма:** Кодты орындап, оның компоненттік құрылымын түсіндіріңіз.

# 1. Пайдаланушы енгізуін қабылдау

def get\_user\_input():

name = input("Аты-жөніңізді енгізіңіз: ")

address = input("Жеткізу мекенжайын енгізіңіз: ")

product = input("Тапсырыс беретін өнімді таңдаңыз: ")

quantity = int(input(f"{product} саны қанша болуы керек? "))

return name, address, product, quantity

# 2. Тапсырысты өңдеу

def process\_order(product, quantity):

product\_prices ={ "Кітап": 1500,"Телефон": 100000, "Ноутбук": 200000}

if product not in product\_prices:

return None, "Өнім қолжетімсіз."

total\_price = product\_prices[product] \* quantity

return total\_price, None

# 3. Тапсырысты растау және шығару

def confirm\_and\_print\_order(name, address, product, quantity, total\_price):

if total\_price is not None:

print(f"\nТапсырыс растау:\n"

f"Аты-жөні: {name}\n"

f"Мекенжай: {address}\n"

f"Өнім: {product} (саны: {quantity})\n"

f"Жалпы баға: {total\_price} тг\n")

print("Тапсырысыңыз сәтті қабылданды!")

else: print(f"\nҚателік: {product} өнімін тапсырыс беру мүмкін емес.")# Бағдарламаның негізгі ағыны

def main():

name, address, product, quantity = get\_user\_input()

total\_price, error = process\_order(product, quantity)

if error:

print(error)

else: confirm\_and\_print\_order(name, address, product, quantity, total\_price)

# Бағдарламаны іске қосу

if \_\_name\_\_ == "\_\_main\_\_": main()

**Тапсырма 2:** пайдаланушы енгізген сан бойынша белгілі бір хабарламаны шығару керек.

**1. Шарттар:**

* 1 → "Сіз пицца таңдадыңыз."
* 2 → "Сіз суши таңдадыңыз."
* 3 → "Сіз бургер таңдадыңыз."
* Басқа мән → "Қате: таңдауды дұрыс енгізіңіз!"

**2. CASE операторының көмегімен бұл процесті іске асырыңыз.**

**Python мысалы (match-case, Python 3.10+):**

python

def order\_choice(choice):

match choice:

case 1:

return "Сіз пицца таңдадыңыз."

case 2:

return "Сіз суши таңдадыңыз."

case 3:

return "Сіз бургер таңдадыңыз."

case \_:

return "Қате: таңдауды дұрыс енгізіңіз!"

# Пайдаланушыдан енгізу алу

try:

user\_choice = int(input("Таңдауыңызды енгізіңіз (1-3): "))

print(order\_choice(user\_choice))

except ValueError:

print("Қате: Сан енгізіңіз!")

# Пайдаланушыдан таңдауды енгізуді сұрау

choice = int(input("1. Пицца\n2. Суши\n3. Бургер\nТаңдауыңызды енгізіңіз: "))

📌 **Тапсырма:**

* Кодты орындаңыз және әртүрлі мәндер енгізіп тексеріңіз.
* CASE операторының артықшылықтарын сипаттаңыз.

**Тапсырма 3:** CASE құралдарын пайдаланып UML диаграммасын жасау

1. **Диаграмма жасау құралының көмегімен (мысалы, draw.io, StarUML)** компоненттік әдіспен жасалған жүйенің UML диаграммасын құрыңыз.
2. **Жүйенің негізгі модульдерін (OrderInput, OrderProcessor, OrderOutput)** көрсетіп, олардың байланысын сипаттаңыз.

📌 **Тапсырма:** UML диаграммасын жасап, оны түсіндіріңіз.