МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

Задание по практической работе №1

по дисциплине «Моделирование программных систем»

Выполнили:
Студент группы ИКБО-66-23 Ковалев А.Э.

Гордильо Ариса Х.Л.

Проверил: Образцов В.М.

2025 г.

Задание 1

Провести оценку качества программного обеспечения и осуществить обоснованный выбор варианта программного обеспечения, применяемого в профессиональной сфере в соответствии с указанной методикой.

1. Осуществить выбор категории анализируемого программного обеспечения.
    - Категория: Категория: Качество предоставляемых услуг , Техническая поддержка, Доступность, Интерфейс, Производительность.

2. Указать характеристики и атрибуты программного обеспечения в соответствии с Национальным стандартом РФ ГОСТ Р ИСО/МЭК 25010-2015 "Информационные технологии. Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Модели качества систем и программных продуктов" (утв. приказом Федерального агентства по техническому регулированию и метрологии от 29 мая 2015 г. N 464-ст).
    - Характеристики: Удовлетворенность, эффективность, защищенность, свобода от риска

3. Определить значения характеристик и атрибутов анализируемого программного обеспечения.

4. Нормализовать указанные данных. Учесть стоимость анализируемого программного обеспечения.

5. Провести анализ методов многокритериального принятия решений (ELECTRE, TOPSIS и т.д.) и выбрать не менее трех из них.

Категории анализируемого программного обеспечения и методы многокритериального принятия решений выбрать самостоятельно.

Таблица 1 — таблица ценности

Веса	**4(0,15)**	**1(0,3)**	**5(0,15)**	**3(0,2)**	**2(0,2)**
Банки/Характеристики	**Качество предоставляемых услуг**	**Техническая поддержка**	**Доступность**	**Интерфейс**	**Производительность**
T	**0,8**	**0,9**	**0,6**	**0,9**	**0,9**
Alfa	**0,85**	**0,8**	**0,6**	**0,75**	**0,9**
VTB	**0,9**	**0,6**	**0,75**	**0,9**	**0,9**
Ozon	**0,7**	**0,8**	**0,9**	**0,8**	**0,7**

После определения характеристик, атрибутов и установления их значений мы приступаем к анализу с использованием методов многокритериального принятия решений (Electre, Soul и Topsis), как показано в таблицах :

Таблица 2 — Результаты ELECTRE анализа

Банк	**Оценка**
T	0.840
Alfa	0.787
VTB	0.787
OZON	0.780

Лучший вариант: T с оценкой 0.840 .

Таблица 3 — Результаты SOUL анализа

Банк	**Оценка**
T	0.717
Alfa	0.637
VTB	0.628
OZON	0.613

Лучший вариант: T с оценкой 0.717.

Таблица 3 — Результаты TOPSIS анализа

Банк	Оценка
T	0.667
Alfa	0.571
VTB	0.534
OZON	0.389

Лучший вариант: T с оценкой 0.667.

Код для метода ELECTRE :

```
import numpy as np

matrix = np.array([
    [0.8, 0.9, 0.6, 0.9, 0.9],
    [0.85, 0.8, 0.6, 0.75, 0.9],
    [0.9, 0.6, 0.75, 0.9, 0.9],
    [0.7, 0.8, 0.9, 0.8, 0.7]
])

weights = [0.15, 0.3, 0.15, 0.2, 0.2]
alternatives = ['T', 'Alfa', 'VTB', 'Ozon']
criteria = ['Качество услуг', 'Тех.поддержка', 'Доступность', 'Интерфейс',
'Производительность']

def electre(matrix, weights):
    results = []
    for i in range(len(matrix)):
        result = []
        for j in range(len(weights)):
            electreRes = matrix[i][j] * weights[j]
            result.append(electreRes)
        results.append(sum(result))
    return results

# Получение результатов
scores = electre(matrix, weights)

# Вывод результатов
results = list(zip(alternatives, scores))
sorted_results = sorted(results, key=lambda x: x[1], reverse=True)
```

```
print("\nРезультаты ELECTRE анализа:")
print("Банк\t\tОценка")
print("-" * 25)
for bank, score in sorted_results:
    print(f"{bank}\t\t{score:.4f}")

# Определение лучшего варианта
best_alternative = sorted_results[0]
print(f"\nЛучший вариант: {best_alternative[0]} с оценкой
{best_alternative[1]:.4f}")
```

Код для метода SOUL :

```
import numpy as np

matrix = np.array([
    [0.8, 0.9, 0.6, 0.9, 0.9],
    [0.85, 0.8, 0.6, 0.75, 0.9],
    [0.9, 0.6, 0.75, 0.9, 0.9],
    [0.7, 0.8, 0.9, 0.8, 0.7]
])

weights = [0.15, 0.3, 0.15, 0.2, 0.2]
alternatives = ['T', 'Alfa', 'VTB', 'Ozon']
criteria = ['Качество услуг', 'Тех.поддержка', 'Доступность', 'Интерфейс',
'Производительность']

def soul(matrix, weights):
    results = []
    for i in range(len(matrix)):
        result = 0
        for j in range(len(weights)):
            # SOUL использует квадратичную функцию полезности
            result += weights[j] * (matrix[i][j] ** 2)
        results.append(result)
    return results

# Получение результатов
scores = soul(matrix, weights)

# Вывод результатов
results = list(zip(alternatives, scores))
sorted_results = sorted(results, key=lambda x: x[1], reverse=True)
```

```
print("\nРезультаты SOUL анализа:")
print("Банк\t\tОценка")
print("-" * 25)
for bank, score in sorted_results:
    print(f"{bank}\t\t{score:.4f}")

# Определение лучшего варианта
best_alternative = sorted_results[0]
print(f"\nЛучший вариант: {best_alternative[0]} с оценкой
{best_alternative[1]:.4f}")
```

Код для метода TOPSIS :

```
import numpy as np

# Исходные данные
alternatives = ['T', 'Alfa', 'VTB', 'Ozon']
criteria = ['Качество услуг', 'Тех.поддержка', 'Доступность', 'Интерфейс',
'Производительность']
weights = [0.15, 0.3, 0.15, 0.2, 0.2]  # веса критериев

# Матрица решений
decision_matrix = np.array([
    [0.8, 0.9, 0.6, 0.9, 0.9],
    [0.85, 0.8, 0.6, 0.75, 0.9],
    [0.9, 0.6, 0.75, 0.9, 0.9],
    [0.7, 0.8, 0.9, 0.8, 0.7]
])

def topsis(matrix, weights):
    # Нормализация матрицы
    normalized = matrix / np.sqrt(np.sum(matrix**2, axis=0))

    # Взвешенная нормализованная матрица
    weighted_normalized = normalized * weights

    # Определение идеального и анти-идеального решения
    ideal_best = np.max(weighted_normalized, axis=0)
    ideal_worst = np.min(weighted_normalized, axis=0)

    # Расчет расстояний до идеального и анти-идеального решения
    s_best = np.sqrt(np.sum((weighted_normalized - ideal_best)**2, axis=1))
    s_worst = np.sqrt(np.sum((weighted_normalized - ideal_worst)**2, axis=1))
```

```
    # Расчет относительной близости к идеальному решению
    performance_score = s_worst / (s_best + s_worst)

    return performance_score

# Получение результатов
scores = topsis(decision_matrix, weights)

# Вывод результатов
results = list(zip(alternatives, scores))
sorted_results = sorted(results, key=lambda x: x[1], reverse=True)

print("\nРезультаты TOPSIS анализа:")
print("Банк\t\tОценка")
print("-" * 25)
for bank, score in sorted_results:
    print(f"{bank}\t\t{score:.4f}")

# Определение лучшего варианта
best_alternative = sorted_results[0]
print(f"\nЛучший вариант: {best_alternative[0]} с оценкой
{best_alternative[1]:.4f}")
```

Выводы:

В данной работе оценка качества программного обеспечения проводилась с использованием методик, основанных на национальном стандарте РФ ГОСТ Р ИСО/МЭК 25010-2015. Были выбраны такие ключевые категории, как качество предоставляемых услуг, техническая поддержка, доступность, интерфейс и производительность, а также определены соответствующие функции и атрибуты для анализа. Используя многокритериальные методы принятия решений (ELECTRE, SOUL и TOPSIS), были оценены различные варианты программного обеспечения, при этом Bank T оказался лучшим вариантом по всем примененным методам, получив наивысшие баллы в каждом анализе (0,840 в ELECTRE, 0,717 в SOUL и 0,667 в TOPSIS).