# МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

# ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

# «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)

Факультет «Информатика и вычислительная техника»

Кафедра «Кибербезопасность информационных систем»

# Лабораторная работа № 1

на тему «Оценка характеристик программ на основе лексического анализа с использованием метрики Холстеда»

|  |
| --- |
| Выполнил: студент группы ВКБ43 |
| Ковалев Данил Петрович |
| (Фамилия, имя, отчество) |
| Проверил: |
| Скляров Алексей Викторович |
| (Фамилия, имя, отчетсво) |

Ростов на-Дону

2025 г.

**Тема работы:** Оценка характеристик программ на основе лексического анализа с использованием метрики Холстеда.

**Цель работы:** изучить методику оценки характеристик программ с использованием метрики Холстеда.

**Ответы на контрольные вопросы**

**Вопрос 1**. Какие метрические характеристики включены в состав измеримых свойств алгоритма (или программы)?

Метрики Холстеда — это набор метрик, разработанных Маурисом Холстедом в 1976 году для оценки сложности программного кода. Эти метрики оценивают различные аспекты программы, включая сложность, объем и трудозатраты на разработку. Основные метрики Холстеда включают:

* – число простых (уникальных) *операторов,* появляющихся в данной реализации;
  + - – число простых (уникальных) операндов, появляющихся в данной реализации;
    - – общее число всех операторов, появляющихся в данной реализации;
    - – общее число всех операндов, появляющихся в данной реализации;
    - – число появлений в программе j-го оператора, где j = 1, 2, 3,…,
    - – число появлений в программе j-го операнда, где j = 1, 2, 3,…, .

Учитывая эти основные метрические характеристики для программы, в конкретной реализации текста программы можно определить:

* + - словарь  **—** это количество уникальных операторов и операндов в программе. Операторы и операнды идентифицируются с использованием токенов (например, ключевых слов, операторов, идентификаторов);
    - длину реализации программы ;
    - длину программы .
    - Объем программы **.** В отличие от длины программы N объем измеряется не количеством слов, а числом двоичных разрядов. Если в словаре имеется слов, то для задания номера любого из них требуется минимум бит.

**Вопрос 2**. Каким условиям удовлетворяют тексты программ?

Тексты программ удовлетворяют условиям:

1. **Отсутствие повторов**: Имена операторов/операндов редко идут подряд.
2. **Циклы вместо дублирования**: Группы операторов не повторяются многократно благодаря использованию циклов.
3. **Процедуры для повторяющихся блоков**: Повторяющиеся фрагменты оформляются как функции/процедуры.
4. **Обязательное использование операндов**: Каждый операнд встречается в тексте хотя бы один раз (неиспользуемые переменные удаляются).

**Вопрос 3**. Каков смысл величины - длина реализации программы?

**Длина реализации** (***N = N1 + N2***) — это общее количество всех операторов и операндов в тексте программы с учетом повторений. Данная метрика позволяет оценить фактический размер кода. Также её можно использовать для выявления несовершенств программирования, которые являются следствием применения не самых удачных приемов программирования.

**Вопрос 4**. В чем измеряется длина программы и объем программы?

Длина программы измеряется количеством слов, объём программы измеряется числом двоичных разрядов.

**Вопрос 5.** Как найти объем программы?

Здесь ***N*** – это длина реализации программы, которая высчитывается путем сложения количества операторов и операндов в программе.

Здесь  **–** это количество уникальных операторов и операндов в программе

**Вопрос 6.** Какие элементы программы относят к словарю операторов?

К операторам относят:

* Арифметические/логические операции (+, -, \*, &&).
* Операторы присваивания (=).
* Управляющие конструкции (if, for, while).
* Вызовы функций/методов (Math.Sin()).
* Разделители (;, ,).
* Парные скобки (), {}, [] (считаются одним оператором).
* Блочные конструкции (begin...end, do...while).

**Вопрос 7**. Что такое потенциальный объем программы? В чем заключается смысл этого понятия?

**Потенциальный объем -** минимально возможный объем программы, реализующей алгоритм в идеальном («потенциальном») языке, где все операции вынесены в готовые функции.

.

Смысл потенциального объема программы заключается в следующем:

Потенциальный объем программы помогает в определении, насколько большим или сложным будет проект, еще до начала активной разработки. Это позволяет команде проекта оценить необходимый объем ресурсов, времени и усилий для успешного выполнения проекта.

**Практическое задание**

**Вариант 6**

**Пункт 1**. Разработать программу, реализующую заданный, в соответствии с вариантом, алгоритм);

Условие моего варианта: написать и протестировать функцию, преобразующую строку восьмеричных цифр в эквивалентное ей целое десятичное число.

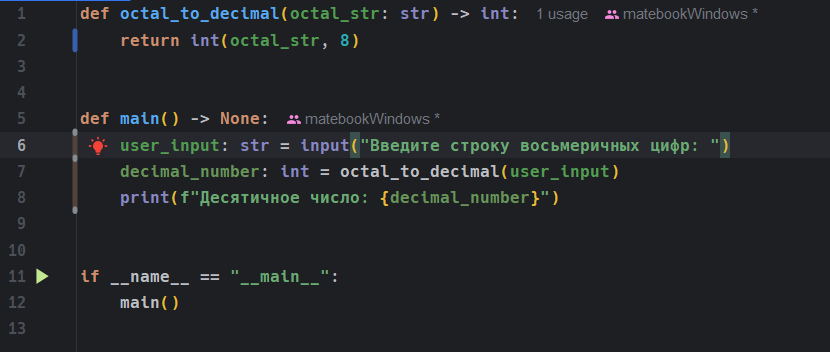


Рисунок 1 – программный код программы

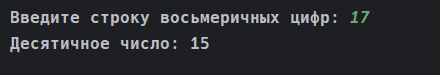


Рисунок 2 – результат выполнения программы

**Пункт 2, 3**. Сформировать словарь программы, охватывающий операнды, а также операторы и операции; словари оформить в виде таких же таблиц, как в рассмотренном примере;

Таблица 1 – операторы, операции в программе

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Операторы,  операции | Номера строк | Количество  повторений |
| 1 | def | 1, 5 | 2 |
| 2 | int | 1,2,7 | 3 |
| 3 | input | 6 | 1 |
| 4 | print | 8 | 1 |
| 5 | Str | 6 | 1 |
| 6 | None | 5 | 1 |
| 7 | If | 11 | 1 |
| 8 | : | 1..2, 5..8, 11…12 | 3 |
| 9 | () | 2, 6, 7, 8 | 4 |
| 10 | {} | 8..8 | 1 |
| 11 | = | 6, 7 | 2 |
| 12 | return | 2 | 1 |
| 13 | “” | 6, 8, 11 | 3 |
| **Всего** | | | **24** |

Таблица 2 – словарь операндов программы

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Операнды | Номера строк | Количество  повторений |
| 1 | octal\_to\_decimal | 1, 7 | 2 |
| 2 | octal\_str | 1, 2 | 2 |
| 3 | 8 | 1 | 1 |
| 5 | main | 5, 12 | 2 |
| 6 | user\_input | 6, 7 | 2 |
| 7 | decimal\_number | 7, 8 | 2 |
| 8 | “Десятичное число: “ | 8 | 1 |
| 9 | “Введите строку восьмеричных цифр” | 6 | 1 |
| 10 | --name-- | 11 | 1 |
| 11 | --main-- | 11 | 1 |
|  |  | **Всего** | **15** |

Таблица 3 – входные и выходные переменные программы

|  |  |
| --- | --- |
| **Входные переменные** | **Выходные переменные** |
| user\_input | «Введите строку восьмеричных цифр: » |
| decimal\_number | «Десятичное число: » |

**Пункт 4**. Рассчитать метрики Холстеда, оформив результат в виде итоговой таблицы;

Таблица 4 – значения метрик Холстеда для программы

|  |  |  |
| --- | --- | --- |
| Наименование  характеристики | Обозначение и формула для вычисления | Значение |
| Число простых (уникальных) операторов и операций |  | 13 |
| Число простых (уникальных) операндов |  | 11 |
| Общее число всех операторов и операций |  | 24 |
| Общее число всех операндов |  | 15 |
| Число входных и выходных переменных (параметров) |  | 4 |
| Словарь программы |  | 24 |
| Длина реализации программы |  | 39 |
| Объем программы (в битах) |  | 178 |
| Потенциальный объем программы |  | 15.5 |
| Уровень реализации программы |  | 0.087 |
| Уровень реализации языка |  | 1.34 |
| Работа программирования |  | 2 045,977 |

**Вывод:** работа программирования высокая, уровень реализации программы маленький, что говорит о том, что программа вполне хорошо написана.

**Задание 2**. Выполнить сравнительную характеристику уже существующей программы.

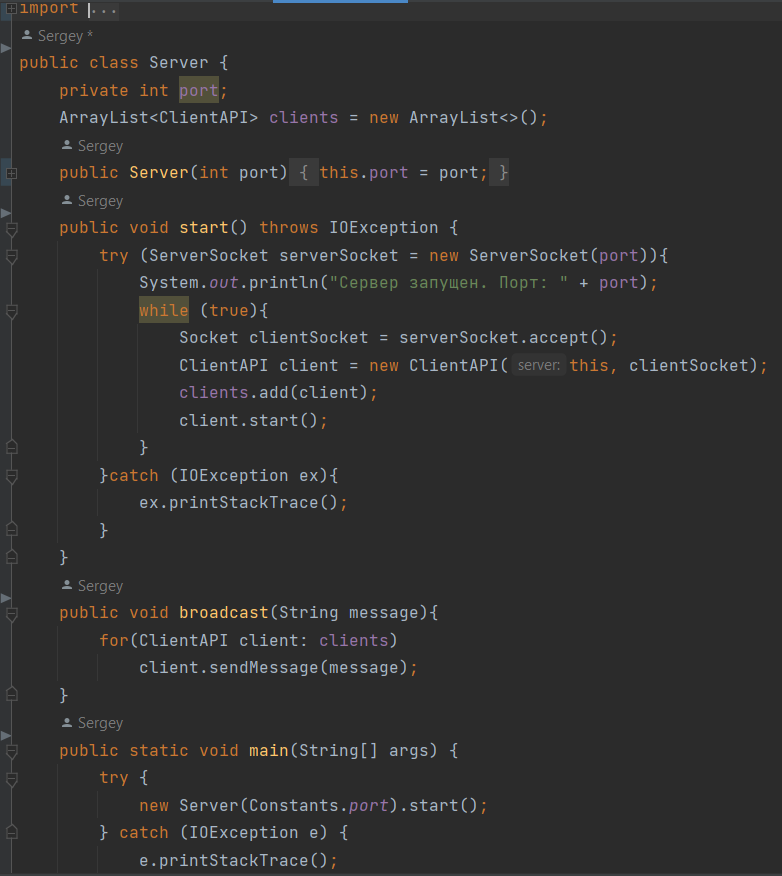


Рисунок 2 – программный код



Рисунок 4 – результат выполнения программы

Таблица 1 – операторы, операции в программе

|  |  |  |
| --- | --- | --- |
| Операторы,  операции | Номера строк | Количество  повторений |
| 1. package | 1 | 1 |
| 1. import | 2,3,4,5,6 | 5 |
| 1. Public class | 7 | 1 |
| 1. Private int | 8 | 1 |
| 1. ArrayList | 9 | 2 |
| 1. Public | 10 | 1 |
| 1. int | 10 | 1 |
| 1. Public void | 13, 26 | 2 |
| 1. throws | 13 | 1 |
| 1. Try…catch | 14…22, 31..33 | 2 |
| 1. ServerSocket | 14, 14 | 2 |
| 1. new | 9, 14,18,32 | 4 |
| 1. System.out.print | 15 | 1 |
| 1. While.. | 16 | 1 |
| 1. Socket | 17 | 1 |
| 1. serverSocket.accept | 17 | 1` |
| 1. ClientAPI | 18,18, 27 | 3 |
| 1. Clients.add | 19 | 1 |
| 1. Clent.start | 20 | 1 |
| 1. IOException | 22,33 | 2 |
| 1. Ex.printStackTrace | 23,34 | 2 |
| 1. String | 26 | 1 |
| 1. Client.sendMessage | 28 | 1 |
| 1. Public static void | 30 | 1 |
| 1. String[] | 30 | 1 |
| 1. Server.start | 32 | 1 |
| 1. For… | 27..28 | 1 |
| 1. “” | 15 | 1 |
| 1. : | 27 | 1 |
| 1. [] | 30 | 1 |
| 1. + | 15 | 1 |
| 1. = | 9, 11,14,17,18 | 5 |
| 1. () | 9,10,13,14,14,15,16,17,18,19,20,22,23,26,27,28,30,32,32,33,34 | 21 |
| 1. <> | 9 | 2 |
| 1. ; | 1,2,3,4,5,6,8,9,11,15,17,18,19,20,23,28,32,34 | 19 |
| 1. {} | 7..37,10..12,13..25,14..22, 16..21,22..24,26..29,30..36,31..33,33..35 | 10 |
| **Всего** | | **103** |

Таблица 2 – словарь операндов программы

|  |  |  |
| --- | --- | --- |
| Операнды | Номера строк | Количество  Повторений |
| 1. server | 1 | 1 |
| 1. utils.Constants | 2 | 1 |
| 1. java.io.\* | 3 | 1 |
| 1. java.net.ServerSocket | 4 | 1 |
| 1. java.net.Socket | 5 | 1 |
| 1. java.util.ArrayList | 6 | 1 |
| 1. Server | 7 | 1 |
| 1. port | 8, 14, 15 | 3 |
| 1. clients | 9, 27 | 2 |
| 1. start | 13 | 1 |
| 1. IOException | 13 | 1 |
| 1. serverSocket | 14 | 1 |
| 1. “Сервер запущен. Порт: ” | 15 | 1 |
| 1. True | 16 | 1 |
| 1. clientSocket | 17, 18 | 2 |
| 1. client | 18, 19, 27 | 3 |
| 1. this | 18 | 1 |
| 1. ex | 22 | 1 |
| 1. broadcast | 26 | 1 |
| 1. message | 26, 28 | 2 |
| 1. main | 30 | 1 |
| 1. args | 30 | 1 |
| 1. Constants.port | 32 | 1 |
| 1. e | 33 | 1 |
| **Итого:** | | 31 |

Таблица 3 – входные и выходные переменные программы

|  |  |
| --- | --- |
| **Входные переменные** | **Выходные переменные** |
|  | «Сервер запущен. Порт: » |

1. рассчитать метрики Холстеда, оформив результат в виде итоговой таблицы;

Таблица 4 – значения метрик Холстеда для программы

|  |  |  |
| --- | --- | --- |
| **Наименование**  **характеристики** | **Обозначение и формула для вычисления** | **Значение** |
| Число простых (уникальных) операторов и операций |  | 36 |
| Число простых (уникальных) операндов |  | 24 |
| Общее число всех операторов и операций |  | 103 |
| Общее число всех операндов |  | 31 |
| Число входных и выходных переменных (параметров) |  | 1 |
| Словарь программы |  | 60 |
| Длина реализации программы |  | 134 |
| Объем программы (в битах) |  | 791,94 |
| Потенциальный объем программы |  | 4,74 |
| Уровень реализации программы |  | 0,006 |
| Уровень реализации языка |  | 0,2844 |
| Работа программирования |  | 131990 |

**Вывод:** уровень реализации программы вполне низкий, что говорит о том, что программа написана нормально.