**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6**

**студента гр. И-1-22**

**Пайвин Кирилл**

По дисциплине: Программная инженерия

Тема работы: Качество программного обеспечения. Метрики, основанные на лексическом анализе программ.

Цель работы: Получить практические навыки оценки качества ПО с использованием метрик лексического анализа.

**Оценка программы по метрике Холтера (Python):**

1) Словарь операндов

|  |  |
| --- | --- |
| Операнд | Количество |
| n | 3 |
| m | 3 |
| self | 4 |
| -3.3 | 1 |
| 3.3 | 1 |
| 0 | 2 |
| 10 | 2 |
| 3 | 2 |
| arr | 1 |
| i | 2 |
| j | 1 |
| matrix | 1 |
| B | 1 |
| random.uniform(-3.3, 3.3) | 1 |

Всего уникальных операндов - 14

2) Словарь операторов

|  |  |
| --- | --- |
| Оператор | Количество |
| = | 8 |
| == | 2 |
| <= | 2 |
| >= | 2 |
| and | 3 |
| or | 1 |
| for | 5 |
| in | 5 |
| break | 1 |
| def | 5 |
| if | 2 |
| while | 1 |
| print | 4 |
| return | 2 |
| class | 1 |

Всего уникальных операторов – 15

3) Словарь операций

|  |  |
| --- | --- |
| Операция | Количество |
| input | 2 |
| append | 1 |
| sum | 1 |
| round | 2 |
| range | 5 |

Всего уникальных операций – 5

n1 = 15 (кол-во уникальных операторов)

n2 = 14 (кол-во уникальных операндов)

N1 = 42 (общее кол-во операторов)

N2 = 25 (общее кол-во операндов)

Объём программы:

V = (N1+N2)\*log2(n1+n2) = 67\*log2(29) = 67\*4,86 = 325,62

Сложность программы:

D = N1/n1 + N2/n2 = 42/15 + 25/14 = 4,58

Длина программы:

L = n1 + n2 = 29

|  |  |  |
| --- | --- | --- |
| Объём программы | Сложность программы | Длина программы |
| 325,62 | 4,58 | 29 |

Потенциальный объём программы = N1+N2 = 67 , реальный объём равен 325. Это указывает на то, что программа имеет достаточно высокий реальный объём, это может свидетельствовать о наличии избыточности в коде.

Предполагаемая длина программы n1\*log2n1 + n2\*log2n2 = 111,7.

Несовершенства могут быть в выборе не совсем подходящей структуры для описания матрицы.

7 строка в коде.

Программа демонстрирует приемлемый уровень, но есть возможности оптимизации и усовершенствования программы.

**Листинг.**

**import** random

**class** **Matrix:**

**def** \_\_init\_\_**(**self**,** n**:** **int,** m**:** **int):**

self**.***n* **=** n

self**.***m* **=** m

self**.***matrix* **=** **[]**

**def** initMatrix**(**self**):**

self**.***matrix* **=** **[[round(**random**.***uniform***(-**3.3**,** 3.3**),** 2**)** **for** \_ **in** **range(**self**.***n***)]** **for** \_ **in** **range(**self**.***m***)]**

**def** printMatrix**(**self**):**

**print(**'\t Matrix:'**)**

**for** i **in** **range(**self**.***m***):**

**for** j **in** **range(**self**.***n***):**

**print(**self**.***matrix***[**i**][**j**],** end**=** ' | '**)**

**print(**''**)**

**def** summRows**(**self**):**

arr **=** **[]**

**for** i **in** **range(**self**.***m***):**

arr**.***append***(round(sum(**self**.***matrix***[**i**]),** 2**))**

**return** arr

**def** main**():**

n **:** **int**

m **:** **int**

**while** **True:**

n **=** **int(input(**"n : "**))**

m **=** **int(input(**"m : "**))**

**if** n **<=** 10 **and** n **>=** 3 **and** m **<=** 10 **and** m **>=** 3**:**

**break**

**elif** n **==** 0 **or** m **==** 0**:**

**return**

A **=** Matrix**(**n**,** m**)**

A**.***initMatrix***()**

B **=** A**.***summRows***()**

A**.***printMatrix***()**

**print(**B**)**

main**()**

**Оценка программы по метрике Холтера (С++):**

1) Список операндов

|  |  |
| --- | --- |
| Операнд | Количество |
| A | 1 |
| B | 1 |
| i | 3 |
| 0 | 11 |
| matrix | 3 |
| rows | 4 |
| columns | 3 |
| 3.3 | 1 |
| -3.3 | 1 |
| a | 1 |
| c | 2 |
| r | 2 |
| m | 2 |
| summ\_all | 1 |
| x | 2 |
| y | 2 |
| 3 | 1 |
| 10 | 1 |
| endl | 4 |

Всего операндов: 19

2) Список операторов

|  |  |
| --- | --- |
| Оператор | Количество |
| struct | 1 |
| new | 4 |
| return | 5 |
| for | 7 |
| do while | 2 |
| + | 2 |
| = | 23 |
| - | 1 |
| == | 2 |
| < | 2 |
| > | 2 |
| if | 2 |
| delete | 1 |
| << | 12 |
| ++ | 7 |
| \* | 3 |

Всего операторов: 16

3) Список операций

|  |  |
| --- | --- |
| Операция | Количество |
| srand | 1 |
| rand | 1 |
| fixed | 1 |
| setprecision | 1 |
| delete[] | 1 |
| new | 4 |
| create\_matrix | 1 |
| init\_matrix | 1 |
| randomFiling | 1 |
| user\_inputNM\_and\_createMatrix | 1 |
| summ\_elem | 1 |
| print\_matrix | 1 |
| clear\_ram | 1 |

n1 = 16 (кол-во уникальных операторов)

n2 = 19 (кол-во уникальных операндов)

N1 = 71 (общее кол-во операторов)

N2 = 46 (общее кол-во операндов)

Объём программы:

V = (N1+N2)\*log2(n1+n2) = 117\*log2(35) = 117\*5,1 = 596,7

Сложность программы:

D = N1/n1 + N2/n2 = 71/16 + 46/19 = 6,8

Длина программы:

L = n1 + n2 = 35

|  |  |  |
| --- | --- | --- |
| Объём программы | Сложность программы | Длина программы |
| 596,7 | 6,8 | 35 |

Потенциальный объём программы = N1+N2 = 117 , реальный объём равен 596,7. Это указывает на то, что программа имеет достаточно высокий реальный объём, это может свидетельствовать о наличии избыточности в коде.

Предполагаемая длина программы n1\*log2n1 + n2\*log2n2 = 143,8.

**Листинг**

#include <iostream>

#include <time.h>

#include <cstdlib>

#include <iomanip>

**using** **namespace** std**;**

struct Matrix

**{**

double**\*** matrix**;**

int rows**,** columns**;**

**};**

Matrix**\*** create\_matrix**(**int rows**,** int cols**){**

Matrix**\*** matrix **=** **new** Matrix**[sizeof(**Matrix**)];**

matrix**->**rows **=** rows**;**

matrix**->**columns **=** cols**;**

matrix**->**matrix **=** **new** double**[**rows **\*** cols**];**

**return** matrix**;**

**}**

Matrix**\*** init\_matrix**(**double**\*** data**,** int rows**,** int cols**){**

Matrix**\*** matrix **=** create\_matrix**(**rows**,** cols**);**

**for(**int i **=** 0**;** i **<** rows**\***cols**;** i**++){**

matrix**->**matrix**[**i**]** **=** data**[**i**];**

**}**

**return** matrix**;**

**}**

double**\*** randomlFiling**(**int r**,** int c**){**

double**\*** a **=** **new** double**[**r**\***c**];**

int max **=** 3.3**,** min **=** **-**3.3**;**

**for(**int i **=** 0**;**i **<** r**\***c**;**i**++){**

a**[**i**]** **=** min **+** **static\_cast<**double**>(**rand**())** **/** **(static\_cast<**double**>(**RAND\_MAX **/** **(**max **-** min**)));**

**}**

**return** a**;**

**}**

Matrix user\_inputNM\_and\_createMatrix**(){**

int r**,** c**;**

**do{**

cout **<<** "Input count rows (3-10).0 - exit: "**;**

cin **>>** r**;**

**if(**r **==** 0**)** exit**(**0**);**

**}while** **(**r **<** 3 **or** r **>** 10**);**

**do{**

cout **<<** "Input count cols (3-10).0 - exit: "**;**

cin **>>** c**;**

**if(**c **==** 0**)** exit**(**0**);**

**}while** **(**c **<** 3 **or** c **>** 10**);**

Matrix**\*** m **=** init\_matrix**(**randomlFiling**(**r**,** c**),** r**,** c**);**

**return** **\***m**;**

**}**

double**\*** summ\_elem**(**Matrix **\***matrix**){**

int rows **=** matrix**->**rows**;**

double**\*** summ\_all **=** **new** double**[**rows**];**

**for(**int x **=** 0**;** x **<** matrix**->**rows**;** x**++){**

**for(**int y **=** 0**;** y **<** matrix**->**columns**;** y**++){**

summ\_all**[**x**]** **+=** matrix**->**matrix**[**x**\*(**matrix**->**columns**)** **+** y**];**

**}**

**}**

**return** summ\_all**;**

**}**

void print\_matrix**(**Matrix**\*** m**){**

cout **<<** fixed **<<** setprecision**(**3**);**

cout **<<**endl **<<** "\t\t" **<<** "Matrix:"**;**

**for(**int x **=** 0**;** x **<** m**->**rows**;** x**++)** **{**

cout **<<** "\n"**;**

**for(**int y **=** 0**;** y **<** m**->**columns**;** y**++)** **{**

cout **<<** "\t" **<<** m**->**matrix**[**x**\*(**m**->**columns**)** **+** y**];**

**}**

**}**

cout **<<** endl**;**

**}**

void clear\_ram**(**Matrix**\*** matrix**){**

**delete** **[]** matrix**;**

**}**

int main**(){**

srand**(**time**(**0**));**

Matrix A **=** user\_inputNM\_and\_createMatrix**();**

print\_matrix**(&**A**);**

cout **<<** endl **<<** "B = [ "**;**

double**\*** B **=** summ\_elem**(&**A**);**

**for(**int i **=** 0**;**i **<** m**.**rows**;**i**++){**

cout **<<** B**[**i**]** **<<** " "**;**

**};**

cout **<<** "]" **<<** endl**;**

system**(**"pause"**);**

**}**

Программа с использованием языка C++ получилась более объемней и сложней по сравнению с программной ,написанной на Python. Но предполагаемая и реальная длина почти сошлась.