Лабораторная работа №2

По курсу «Основы программной инженерии»

«Оценка качества программного продукта»

Цель: изучение основных методов и подходов оценки качества программного продукта.

Задачи:

- изучить основы метрической теории Холстеда;
- для написанных ранее программ произвести расчет количественных характеристик программ;
- сравнить полученные результаты.

Содержание отчета:

1. Титульный лист;
2. Цель, задачи работы;
3. Листинг программ;
4. Результат сравнения
5. Вывод

Основные теоретические сведения

Метрики Холстеда предлагают разумный подход к решению следующих задач:

-предсказание условий, необходимых для программирования по предложенным проектам;

-определение норм первоначальных ошибок;

-количественная оценка языков программирования и эффекта модульности;

-обоснование метода измерения различий между программами, написанными специалистами разного уровня.

В основе вычисления метрик Холстеда лежит концепция, согласно которой алгоритм состоит только из операторов и операндов (проверяется рассмотрением простых вычислительных машин с форматом команд, содержащим две части: код операции и адрес операнда). Операнды -переменные или константы, используемые в данной реализации алгоритма.

Операторы-комбинации символов, влияющие на значение или порядок операндов.

В основе вычисляемых свойств алгоритма лежат следующие характеристики:

-η_1 -число различных операторов данной реализации;

-η_2 -число различных операндов данной реализации;

-N_1 -общее число всех операторов;

-N_2 -общее число всех операндов;

-n2*-число различных входных и выходных операндов.

На основании приведенных выше характеристик вычисляются:

-словарь $n = \eta_1 + \eta_2$;

-длина реализации $N = N_1 + N_2$.

Метрики Холстеда включают следующие характеристики:

1) Длина программы: N'= η_1 * $\log_2 \eta_1$ + η_1 * $\log_2 \eta_2$

Если программа состоит из нескольких модулей (m -число модулей), то для каждого модуля определяется η_1 среднее ($\eta_{1ср}$) и η_2 среднее ($\eta_{2ср}$). В этом случае длина программы

N' = m*(η_1ср* log2($\eta_{1ср}$) + η_2 ср* log2($\eta_{2ср}$)).

2) Объем программы: V = N * log2(η).

Такая интерпретация дает объем программы в битах. Объем зависит от языка программирования, на котором реализован алгоритм.

3) Потенциальный (минимальный) объем: V* = (2+ η_2*) * log2(2+ η_2*).

Минимально возможный объем предполагает существование языка, в котором действия, выполняемые индивидуальным модулем, уже определены или реализованы, возможно, в виде процедуры или функции.

4) Граничный объем: V'' = (2+ (η_2')* log2(η_2')) * log2(2+η_2').

5) Соотношения между операциями и операндами (зависимость числа операндов n2 от числа операций n1: A = η_2'/(η_2'+2) * log2(η_2'/2)

B = η_2' -2 * A

η_2 = A * η_1 + B

-это частота использования операндов.

6) Уровень программы: L = V'/ V,

где V' -потенциальный объем, V -объем программы.

L=1 для потенциального языка, в котором присутствует любая процедура, которая могла бы понадобиться (число таких процедур близко к бесконечности).

Альтернативное определение уровня L: L' = (η_1') * η_2 /(η_1*N2), где η_1' =2.

7) Интеллектуальное содержание: I = L' * V или: I = 2 * η_2 /(η_1 * N2)*N* log2(η).

Интеллектуальное содержание - мера того, "сколько было сказано в программе", зависит от сложности задачи. (I≈11-13).

8) Работа по программированию (общее число элементарных мысленных различий, требуемых для порождения программы): E= V /L или E = V^2 / V',где E –общее число элементарных умственных различений, требуемых для порождения программы.

Это умственная работа, затрачиваемая на превращение заранее разработанного алгоритма в фактическую реализацию на языке программирования.

Правильное разбиение на модули уменьшает работу по программированию:

E = E1 + E2 + E3 + ...

9) Приближенное время программирования: T' = E/S, где S = 18 моментов (различий)/секунд

-постоянная Страуда.

Момент - время, требуемое человеческому мозгу для выполнения элементарных различений. Альтернативное определение времени T: T' = η_1*N2*(η_1 * log2(η_1)+ η_2 * log2(η_2))/(2*S*η_2).

10) Уровень языка: $\lambda = L^2 \times V$.

Уровень языка определяет его производительность.

11) Уравнение ошибок:

Число переданных ошибок в программе: B = V/E0,

где E0 = $V' * V' * V'$ /(A x A) -среднее число элементарных различений между возможными ошибками в программировании.

«Переданные ошибки» -ошибки, остающиеся после отладки модуля.

Выполнение лабораторной работы:

1. Для индивидуального модуля определить характеристики программы

(η_1, η_2, η, N1, N2, N, η_2').

2. Рассчитать метрики Холстеда по формулам 1-11.

3. Оценить качество реализации алгоритма на основании метрик Холстеда.

4. Оформить отчет.

Пример:

Ниже представлен листинг программ:

Pascal

```
program lab1;
type
  ary = array [1..10] of real;
procedure sort(var a: ary; n: integer);
var
  no_change: boolean;
  j: integer;
  procedure swap(var p, q: real);
  var
    hold: real;
  begin
    hold := p;
    p := q;
    q := hold
  end;
begin
  repeat
    no_change := true;
    for j := 1 to n - 1 do
    begin
      if a[j] > a[j + 1] then
      begin
        swap(a[j], a[j + 1]);
        no_change := false
      end
    end
  until no_change
end;

var
  mass: ary;
  i: integer;

begin
  for i := 1 to 10 do mass[11 - i] :=
i;
  sort(mass, 10);
```

C

```
typedef float ary[10];
void swap(float &p,float &q)
 {
  float hold = p;
  p = q;
  q = hold;
 }
void sort(ary a, int n)
 {
  int change;
    do {
        change = 0;
        for (int j=0; j<="" p="">
           {
              if (a[j] > a[j+1])
              {
                  swap(a[j],a[j+1]);
                  change=1;
              }
           }
       }
    while (change);
}
void main()
{
      ary mass;
      for (int i=0; i<10; i++) mass[9-i] = i+1;
      sort(mass,10);
}
```

`end.`	

Измерение свойств алгоритмов:

- **Pascal**

Операторы			Операнды		
Номер	Оператор	Число вхождений	Номер	Оператор	Число вхождений
1	Begin end	5	**1**	1	6
2	+	2	**2**	10	3
3	-	2	**3**	11	1
4	;	17	**4**	a	5
5	:=	8	**5**	i	4
6	>	1	**6**	j	6
7	[]	5	**7**	n	2
8	for	2	**8**	p	3
9	if	1	**9**	q	3
10	program	1	**10**	true	1
11	repeat	1	**11**	false	1
12	sort	2	**12**	no_change	4
13	swap	2	**13**	mass	3
14	type	1	**14**	lab1	1
15	array	1	**15**	hold	3
			16	ary	1
Итог 51			**Итог 47**		

- **C**

Операторы			Операнды		
Номер	Оператор	Число вхождений	Номер	Оператор	Число вхождений
1	() или {}	7	**1**	0	3
2	+	3	**2**	1	5
3	-	2	**3**	9	1
4	;	16	**4**	10	3
5	=	8	**5**	i	5
6	>	1	**6**	j	7
7	[]	5	**7**	q	3
8	++	2	**8**	p	3
9	<	2	**9**	a	5
10	do while	1	**10**	n	2
11	for	2	**11**	mass	3
12	if	1	**12**	hold	2
13	main	1	**13**	change	4
14	sort	2			
15	swap	2			
16	typedef	1			
17	ary[]	1			
18	&	2			
Итог 59			**Итог 46**		

Расчетные характеристики:

Название	**Pascal**	**C**	**Формула**
Число уникальных операторов (n1)	15	18	
Число уникальных операндов (n2):	16	13	
Общее число операторов (N1):	51	59	
Общее число операндов (N2):	47	46	
словарь программы (n):	31	31	$\eta = \eta_1 + \eta_2$
Экспериментальная длина программы (Nэ):	98	105	Nэ= N1+ N2
Теоретическая длина программы ($\hat{N}$):	122.6	123.16	$\hat{N} = \eta_1 \log_2\eta_1 + \eta_2 \log_2\eta_2$
Объём программы (V):	485.51	520.19	$V = N_э \log_2 n$
Потенциальный объём (V*):	15.51	15.51	$V^* = (2_+ \eta_2^*) \log_2(2_+ \eta_2^*)$ (η_2=4 – количество параметров)
Граничный объем (Vгр)	25.85	25. 85	$N_{гр} = \eta_1^* \log_2 \eta_1^* + \eta_2^* \log_2 \eta_2^* = 2 + \eta_2^* \log_2 \eta_2^*$ $V_{гр} = N_{гр} \log_2 \eta^* = (2 + \eta_2^* \log_2 \eta_2^*) \log_2 (2 + \eta_2^*)$
Уровень программы (L):	0.032	0.03	$L = \frac{V^*}{V}$
Сложность программы (S):	31.3	33.54	$S = \frac{1}{L}$
Оценка уровня программы (L^):	0.045	0.031	$\hat{L} \cong \frac{2}{\eta_1}\frac{\eta_2}{N_2}$
Интеллект программы (I):	22.04	16.33	$I = \frac{2}{\eta_1}\frac{\eta_2}{N_2} \times (N_1+N_2) \log_2(\eta_1+\eta_2)$
Работа по программированию (E):	15198	17446	$E = \frac{V^2}{V^*}$
Время программирования (T):	1520	1745	$T = \frac{E}{S}$ (S=10 – число страудовских «моментов» в секунду)
Ожидание времени кодирования (T^):	1338	1943	$\hat{T} = \frac{\eta_1 N_2 (\eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2) \log_2 \eta}{2\eta_2 S}$

Уровень языка программирования (λ):	0.495	0.462	$\lambda = L^2 \times V^*$
Ожидаемое число ошибок (B)	0.16	0.17	B= V / 3000

Контрольные вопросы:

1. Где можно использовать метрики Холстеда?

2. Чем определяются характеристики программы?

3. Как оценить качество реализации алгоритма по метрикам?