ПРАКТИЧЕСКАЯ РАБОТА № 3.

Тема: метрики сложности потока управления программ.

задание 1) Рассчитать метрику Маккейба (цикломатическое число графа программы).

задание 2) Рассчитать метрику Джилба (насыщенность программы условными операторами и максимальный уровень их вложенности).

задание 3) Рассчитать метрику граничных значений (скорректированную сложность вершин графа программы).

Приложение. Методические рекомендации по выполнению заданий.

Метрики сложности потока управления программ принято определять на основе представления программ в виде управляющего ориентированного графа *G=(V, E)*, где *V* — вершины, соответствующие операторам, а *E* — дуги, соответствующие переходам между операторами. В дуге *(v, u)* вершина *v* является исходной, а *u* — конечной. При этом *u* непосредственно следует за *v,* а *v* непосредственно предшествует *u.* Если путь от *v* до *u* состоит более чем из одной дуги, тогда *u* следует за *v,* а *v* предшествует *u.*

Частным случаем представления ориентированного графа программы можно считать детализированную схему алгоритма, построенную в соответствии с положениями стандарта ГОСТ 19.701-90. Аналогами вершин графа являются блоки алгоритма, причем данные блоки имеют разное графическое представление в зависимости от их назначения. Дугам графа соответствуют линии передачи управления между блоками алгоритма.

МЕТРИКА МАККЕЙБА (цикломатическая сложность графа программы, цикломатическое число Маккейба) предназначена для оценки трудоемкости тестирования программы. Метрика Маккейба определяет минимальное количество тестовых прогонов программы, необходимых для тестирования всех ее ветвей (разветвлений).

Данная метрика определяется по формуле: $\boldsymbol{Z(G) = e - \upsilon + 2p}$, где *e* — число дуг ориентированного графа *G;* υ — число вершин; *p* — число компонентов связности графа.

Число компонентов связности графа – это количество дуг, которые необходимо добавить для преобразования графа в сильносвязный. Сильносвязным называется граф, любые две вершины которого взаимно достижимы. Для корректных программ, не имеющих недостижимых от начала программы участков и «висячих» точек входа и выхода, сильносвязный граф получается путем соединения дугой вершины, обозначающей конец программы, с вершиной, обозначающей начало этой программы.

Рассчитаем метрику Маккейба для программы, схема алгоритма которой приведена на Рисунке 1: на 1.(а) приведена блок-схема алгоритма по ГОСТ, а на 1.(б) – граф данной программы. Действия, выполняемые блоками программы, в примере не показаны на схеме (это не имеет значения для метрики). Внутри каждого блока помещены их номера. Компонент связности графа обозначен штриховой дугой. Число дуг $e = 8$, число вершин $\upsilon = 7$, $p = 1$.

Цикломатическое число Маккейба равно $\boldsymbol{Z(G)} = 8 - 7 + 2*1 = 3$.

Значение метрики Маккейба показывает, что в схеме алгоритма (Рисунок 1) можно выделить три базисных независимых пути (называемых еще линейно независимыми контурами):

<u>1-й путь</u>. 1 – 2 (нет) – 4 (нет) – 5 – 7.

<u>2-й путь</u>. 1 – 2 (да) – 3 – 2 (нет) – 4 (нет) – 5 – 7.

<u>3-й путь</u>. 1 – 2 (нет) – 4 (да) – 6 – 7.

Вторым возможным вариантом совокупности базисных независимых путей является:

<u>1-й путь</u>. 1 – 2 (да) – 3 – 2 (нет) – 4 (нет) – 5 – 7.

<u>2-й путь</u>. 1 – 2 (нет) – 4 (нет) – 5 – 7.

<u>3-й путь</u>. 1 – 2 (да) – 3 – 2 (нет) – 4 (да) – 6 – 7.

Таким образом, для тестирования совокупности базисных независимых путей исследуемой программы необходимо выполнить минимально три тестовых прогона.

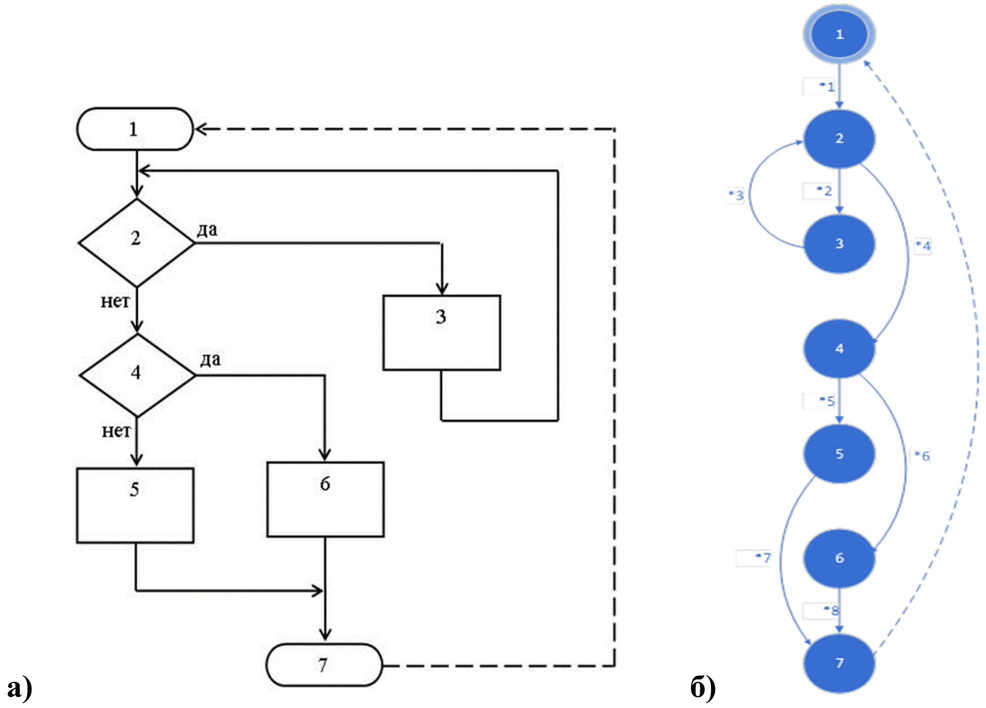


Рисунок 1. Пример алгоритма программы:
а) схема по ГОСТ; б) граф программы

МЕТРИКА ДЖИЛБА определяет логическую сложность программы как насыщенность программы условными операторами (операторами, которые содержат логические выражения).

Используются следующие характеристики метрики Джилба: *CL* — количество условных операторов, характеризующее абсолютную сложность программы; *cl* — насыщенность программы условными операторами, характеризующая относительную сложность программы; *cl* определяется как отношение *CL* к общему количеству операторов программы (здесь под операторами подразумеваются узлы графа, кроме начального и конечного). Условные операторы в графе программы представлены узлами отбора. Расширением метрики Джилба является *максимальный уровень вложенности условного оператора CLI*.

Использование в программе оператора выбора (например, CASE) с *n* разветвлениями эквивалентно применению *n*–1 оператора IF–THEN–ELSE с глубиной вложенности *n*–2.

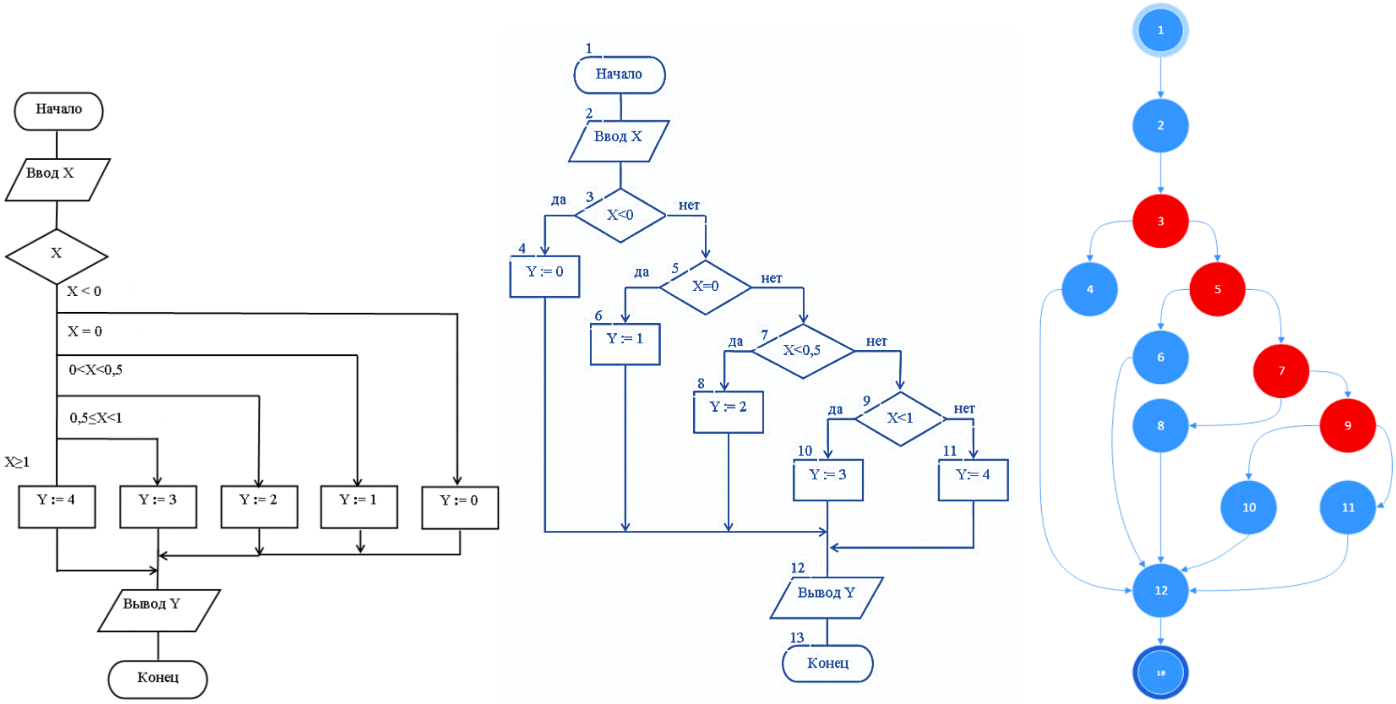
Значение метрики Джилба для программы, алгоритм которой представлен на **Рисунке 1**:

CL = 2 (вершины отбора №2 и №4), ***cl*** = 0,4 (общее число операторов программы равно 5, т.е. 7 узлов за вычетом начальной и конечной вершины), ***CLI*** = *0* (поскольку вершины отбора идут последовательно).

На рисунке 2 приведены схемы разветвляющегося алгоритма некоторой программы. На схеме 2.(а) используется выбор, обозначаемый символом «Решение» с пятью разветвлениями (*n*=5). Эквивалентный алгоритм вычисления той же программы, использующий несколько операторов IF–THEN–ELSE, представлен на схеме 2.(б). На данной схеме число операторов IF–THEN–ELSE равно четырем (*n*–1), а максимальный уровень вложенности оператора IF–THEN–ELSE равен трем (*n*–2). Это можно наблюдать из графа программы (рисунок 2.(в)), на изображении которого узлы ветвления (узлы отбора) обозначены красным цветом.

Таким образом, для схем вариантов алгоритма программы на **Рисунке 2:**

CL = 4, ***cl*** = 0,36 (общее число операторов программы равно 11, т.е. 13 узлов за вычетом начальной и конечной вершины), ***CLI*** = 3 (три вершины отбора вложены одна в другую: вершина 5 вложена в 3, вершина 7 вложена в 5, а 9 в свою очередь – в 7).



(а) **(б)** **(в)**

Рисунок 2. Схемы разветвляющегося алгоритма программы:
а) используется символ «Решение» с многими выходами;
б) используется символ «Решение» с двумя выходами;
в) граф программы.

Значения метрики Маккейба для алгоритмов на Рисунке 2 также совпадают, что показывает их эквивалентность.

Для схемы алгоритма 2.(а): $Z(G) = 13 - 10 + 2 = 5$.

Для схемы алгоритма 2.(б): $Z(G) = 16 - 13 + 2 = 5$.

Следует отметить, что сложность программы с помощью метрики Джилба не всегда возможно посчитать на основе схемы алгоритма, так как схема алгоритма может быть представлена укрупнено. Поэтому значения метрики Джилба в общем случае следует определять на основе ее исходного текста или детализированной схемы алгоритма, каждый блок которой содержит один оператор программы.

МЕТРИКА ГРАНИЧНЫХ ЗНАЧЕНИЙ базируется на определении скорректированной сложности вершин графа программы. Пусть $G=(V, E)$ – ориентированный граф программы с единственной начальной и единственной конечной вершинами. Число входящих в вершину дуг называется *отрицательной степенью вершины*, а число исходящих из вершины дуг — *положительной степенью вершины*. С учетом этого набор вершин графа можно разбить на две группы:

– *принимающие вершины*, у которых положительная степень меньше или равна 1;
– *вершины выбора* (*предикатные вершины, условные вершины, вершины отбора*), у которых положительная степень больше или равна 2.

Для оценки сложности программы с использованием метрики граничных значений граф G разбивается на максимальное число подграфов**,** удовлетворяющих следующим условиям:

– вход в подграф осуществляется через вершину выбора;
– каждый подграф включает вершину (нижнюю границу подграфа), в которую можно попасть из любой другой вершины подграфа.

Скорректированная сложность вершины выбора равна числу вершин в её подграфе.

Каждая принимающая вершина имеет скорректированную сложность равную 1.

Конечная вершина имеет скорректированную сложность равную 0.

Абсолютная граничная сложность программы S_a определяется как сумма скорректированных сложностей всех вершин графа *G*.

Относительная граничная сложность программы S_o определяется по формуле: $\boldsymbol{S_o = 1 - \frac{v-1}{S_a}}$, где S_a — абсолютная граничная сложность программы; v – общее число вершин графа программы.

Таблица 1 содержит описание подграфов программы, граф которой представлен на Рисунке 1.б.
Таблица 2 содержит скорректированные сложности всех вершин графа этой программы.
Вершина отбора, которая содержит петлю, входит в свой подграф (например, вершина 2 на рисунке 1.б).

Таблица 5 – **Свойства подграфов программы** (Рисунок 1.б)

	Номер вершины выбора	
	2	**4**
Номера вершин перехода	3, 4	5, 6
Номер нижней границы подграфа	4	7
Номера вершин подграфа	2, 3, 4	5, 6, 7
Скорректированная сложность вершины выбора	3	3

Таблица 6 – **Скорректированные сложности вершин графа программы** (Рисунок 1.б)

Номер вершины графа программы	1	2	3	4	5	6	7	
Скорректированная сложность вершины графа	1	3	1	3	1	1	0	$\boldsymbol{S_a = 10}$

Абсолютная граничная сложность программы на Рисунке 1.(б) равна $S_a = 10$.
Относительная граничная сложность данной программы равна $S_o = 1 - (7 - 1)/10 = 0{,}4$.

В Таблице 3 представлено описание подграфов программы, граф которой изображен на Рисунке 2.в. Таблица 4 содержит скорректированные сложности всех вершин графа программы.

Таблица 3 – **Свойства подграфов программы** (Рисунок 2.в)

	Номер вершины выбора			
	3	**5**	**7**	**9**
Номера вершин перехода	4, 5	6, 7	8, 9	10, 11
Номер нижней границы подграфа	12	12	12	12
Номера вершин подграфа	4, 5, 6, 7, 8, 9, 10, 11, 12	6, 7, 8, 9, 10, 11, 12	8, 9, 10, 11, 12	10, 11, 12
Скорректированная сложность вершины выбора (мощность подграфа)	9	7	5	3

Таблица 4 – **Скорректированные сложности вершин графа программы** (Рисунок 2.в)

Номер вершины графа программы	1	2	3	4	5	6	7	8	9	10	11	12	13	
Скорректированная сложность вершины графа	1	1	9	1	7	1	5	1	3	1	1	1	0	$\boldsymbol{S_a = 32}$

Таким образом, абсолютная граничная сложность программы на Рисунке 2.(в) равна $S_a = 32$.
Относительная граничная сложность данной программы равна $S_o = 1 - (13 - 1)/32 = 0{,}625$.