БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра Информатики

Факультет компьютерных систем и систем

Специальность «Информатика и технологии программирования»

Индивидуальная практическая работа

по дисциплине «Метрология, стандартизация и сертификация в информационных технологиях»

Вариант № 30

Выполнил студент гр.353501: Губский М.Д.

Минск 2014

**Условие задания**

Ввести массив А (7, 8). Найти сумму элементов каждого столбца, максимальную и минимальную из этих сумм. Вывести массив, полученные суммы, номера столбцов, где находятся максимальная и минимальная суммы.

**Листинг кода программы**

#include "stdafx.h"

#include <stdlib.h>

#include <time.h>

int summ (int i, int\*\* a) //нахождение суммы столбца

{

int sum=0;

for (int k = 0; k < 7; k++)

{

sum=sum+a[k][i];

}

return sum;

}

int max (int\*b) //нахождение максимального столбца

{

int max=0;

for(int i=1; i<8;i++)

{

if (b[max]<b[i])

{

max=i;

}

}

return max+1;

}

int min (int\*b) //нахождение минимального столбца

{

int min=0;

for(int i=1; i<8;i++)

{

if (b[min]>b[i])

{

min=i;

}

}

return min+1;

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

srand(time(NULL));

int\*\* a=(int\*\*)malloc(7\*sizeof(int\*)); //выделение памяти под массив 7х8

for (int i = 0; i < 7; i++)

{

a[i]=(int\*) malloc(8\*sizeof(int));

}

for (int i = 0; i < 7; i++) //ввод массива 7х8

{

for (int j = 0; j < 8; j++)

{

a[i][j]=rand()%30;

printf("%d ",a[i][j]);

}

printf("\n");

}

int\* b=(int\*)malloc(8\*sizeof(int)) ; //выделение памяти под массив 1х8

for (int i = 0; i < 8; i++) //заполнение суммами массива 1х8

{

b[i]=summ(i, a);

}

printf("\n");

for (int j = 0; j < 8; j++) //вывод массива 1х8

{

printf("%d ",b[j]);

}

printf("\n min summ from %d, max summ from %d cols\n ",min(b), max(b));

system("pause");

return 0;

}

**Метрики размера программ. Метрики Холстеда**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***j*** | **Оператор** | ***f1j*** | ***i*** | **Операнд** | ***f2i*** |
|  | ; | 38 |  | A | 8 |
|  | = | 19 |  | B | 8 |
|  | \* | 16 |  | i | 27 |
|  | / | 6 |  | j | 9 |
|  | ( ) | 6 |  | k | 5 |
|  | + | 3 |  | min | 7 |
|  | ++ | 8 | 7. | sum | 4 |
|  | {} | 14 | 8. | max | 7 |
|  | if() | 2 | 9. | 30 | 1 |
|  | for() | 9 | 10. | 0 | 9 |
|  | < | 9 | 11. | 1 | 4 |
|  | **>** | 1 | 12. | 7 | 4 |
|  | printf() | 5 | 13. | 8 | 7 |
|  | “ ” | 9 |  |  |  |
|  | % | 5 |  |  |  |
|  | [] | 14 |  |  |  |
|  | , | 8 |  |  |  |
|  | malloc() | 3 |  |  |  |
|  | summ() | 2 |  |  |  |
|  | min() | 2 |  |  |  |
|  | max() | 2 |  |  |  |
|  | system() | 1 |  |  |  |
|  | srand() | 1 |  |  |  |
|  | rand() | 1 |  |  |  |
|  | time() | 1 |  |  |  |
|  | sizeof() | 1 |  |  |  |
| **η1 *= 26*** |  | ***N1 =189*** | **η2 *=13*** |  | ***N2 = 100*** |

Словарь программы **η**=26 + 13 = 39.

Длина программы ***N*** = 189 + 100 = 289.

Объем программы ***V*** = =1527.

**Детализированная схема алгоритма**





**Метрики сложности потока управления программ**

***Метрика Маккейба*** (цикломатическая сложность графа программы, цикломатическое число Маккейба) предназначена для оценки трудоемкости тестирования программы. Данная метрика определяется по формуле:

***Z(G) = e – ʋ + 2p*,**

где *е —* число дуг ориентированного графа *G; ʋ —* число вершин; *р —* число компонентов связности графа.

Для схемы алгоритма, приведенной в данной работе, число дуг *е* = 44, число вершин *ʋ* = 36*,* число компонент связности *р* = 1. Компонент связности графа обозначен штриховой дугой. Цикломатическое число Маккейба равно *Z(G)* = 44 – 36 + 2 = 10.

Значение метрики Маккейба показывает, что в схеме алгоритма можно выделить 10 базисных независимых путей (называемых также линейно независимыми контурами):

1-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(да) – 21 – 22 – 20 (нет) – 23 – 24 – 25 (да) – 26 (да) – 27 – 28 – 25 (нет) – 29 –30 – 31 (да) – 32(да) – 33 – 34 – 31 (нет) – 35 – 36.

2-й путь. 1 – 2 – 3 (нет) – 10 – 11 (нет) – 19 – 20(нет) – 23 – 24 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

3-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(нет) – 23 – 24 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

4-й путь. 1 – 2 – 3 (да) – 4 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(нет) – 23 – 24 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

5-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(да) – 21 – 22 – 20 (нет) – 23 – 24 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

6-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(нет) – 23 – 24 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

7-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(да) – 21 – 22 – 20 (нет) – 23 – 24 – 25 (да) – 26 (да) – 27 – 28 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

8-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(да) – 21 – 22 – 20 (нет) – 23 – 24 – 25 (да) – 26 (нет) – 28 – 25 (нет) – 29 –30 – 31 (нет) – 35 – 36.

9-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(да) – 21 – 22 – 20 (нет) – 23 – 24 – 25 (да) – 26 (да) – 27 – 28 – 25 (нет) – 29 –30 – 31 (да) – 32(да) – 33 – 34 – 31 (нет) – 35 – 36.

10-й путь. 1 – 2 – 3 (да) – 4 – 5 (да) – 6 – 8 – 9 – 5 (нет) – 7 – 3 (нет) – 10 – 11 (да) – 12 – 13 – 14 (да) – 15 – 16 – 14(нет) – 17 – 18 – 11 (нет) – 19 – 20(да) – 21 – 22 – 20 (нет) – 23 – 24 – 25 (да) – 26 (нет) – 28 – 25 (нет) – 29 –30 – 31 (да) – 32(да) – 33 – 34 – 31 (нет) – 35 – 36.

Таким образом, для тестирования совокупности базисных независимых путей исследуемой программы необходимо выполнить минимально 10 тестовых прогонов.

***Метрика Джилба*** определяет логическую сложность программы как насыщенность программы условными операторами IF–THEN–ELSE

Для данной схемы алгоритма *CL* = 9, *cl =* 0,25 (количество операторов программы равно 36). Максимальный уровень вложенности условных операторов  *CLI =* 1.

**Метрика граничных значений**

**Свойства подграфов программы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Свойства подграфов программы | Номер вершины выбора | | | | |
| 3 | 5 | 11 | 14 | 20 |
| Номера вершин перехода | 4, 10 | 6, 9 | 12, 19 | 15, 17 | 21, 23 |
| Скорректированная сложность вершины выбора | 8 | 5 | 9 | 4 | 4 |
| Номера вершин подграфа | 3,4, 5, 6, 7, 8,9 | 5,6,7,8 | 11,12, 13, 14, 15, 16, 17, 18 | 14,15, 16 | 20,21, 22 |
| Номер нижней границы подграфа | 10 | 3 | 19 | 17 | 23 |

**Свойства подграфов программы (продолжение)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Свойства подграфов программы | Номер вершины выбора | | | |
| 25 | 26 | 31 | 32 |
| Номера вершин перехода | 26, 29 | 27, 28 | 32, 35 | 33, 34 |
| Скорректированная сложность вершины выбора | 5 | 2 | 5 | 2 |
| Номера вершин подграфа | 25,26,27,28 | 27 | 31,32,33, 34 | 33 |
| Номер нижней границы подграфа | 29 | 28 | 35 | 34 |

**Скорректированные сложности вершин графа программы**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер вершины графа программы |  |  |  |  |  |  |  |  |  |  |  |  |  | 14 | 15 | 16 | 17 | 18 |
| Скорректированная сложность вершины графа | 1 | 1 | 8 | 1 | 5 | 1 | 1 | 1 | 1 | 1 | 9 | 1 | 1 | 4 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | ***Sa*** |
| 1 | 4 | 1 | 1 | 1 | 1 | 5 | 2 | 1 | 1 | 1 | 1 | 5 | 2 | 1 | 1 | 1 | 0 | **70** |

Таким образом, абсолютная граничная сложность *Sa* программы, схема алгоритма которой приведена, равна 70. Относительная граничная сложность данной программы равна

*So*= l – (36 – 1)/70 = 0, 5

Следующая таблица содержит метрики сложности потока управления для программы.

**Метрики сложности потока управления программ**

|  |  |
| --- | --- |
| Метрики сложности потока управления | |
| Метрика Маккейба *Z(G)* | 10 |
| Абсолютная сложность программы *CL* по метрике Джилба | 9 |
| Относительная сложность программы *cl* по метрике Джилба | 0,25 |
| Максимальный уровень вложенности условного оператора *CLI* по метрике Джилба | 1 |
| Метрика граничных значений (абсолютная граничная сложность программы) *Sa* | 70 |
| Метрика граничных значений (относительная граничная сложность программы) *S0* | 0,5 |

**Спен программы**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Идентификатор | A[][] | B[] | i | j | k | min | max | sum | Суммарный спен программы |
| Спен | 8 | 8 | 27 | 9 | 5 | 7 | 7 | 4 | 75 |

Идентификаторы *A[][], B[], i, j, p, min, max, sum* инициализируются и используются в теле программы, поэтому их спен равен (*f2i* ) в метриках Холстеда.

**Метрики Чепина программы**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Полная метрика Чепина | | | | Метрика Чепина ввода/вывода | | | |
| Группа переменных | Р | М | С | Т | Р | М | С | Т |
| Переменные, относящиеся к группе | A[][] | sum | i,j,k, B[], min,max | -- | A[][] | sum | B[],  min,max | -- |
| Количество переменных в группе | р = 1 | m = 1 | c = 6 | t = 0 | р = 1 | m=1 | c=3 | t=0 |
| Метрика Чепина | Q = 1\*1 + 2\*1 + 3\*6 + 0,5\*0 = 21 | | | | Q = 1\*1 + 2\*1 + 3\*3 + 0,5\*0 = 12 | | | |

Переменная A[][] является вводимой переменной для задания массива элементов 7х8. Идентификатор sum создается внутри программы и не являются управляющими переменными. Идентификаторы *i,* *j, k, B[], min, max* являются счетчиками в циклах, *B[]* используется в сравнении элементов массива, min, max-индексы элементов массива, также используются в сравнении элементов массива.Эти идентификаторы участвуют в управлении работой программного модуля. В список переменных ввода/вывода программы входят переменные *A[][], sum, B[], min, max.*