*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана***  ***(национальный исследовательский университет)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ \_\_\_\_\_ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА \_\_\_\_\_\_\_ Компьютерные Системы и сети (ИУ6)\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по лабораторной работе № 3**

**Вариант 8**

**Дисциплина: технология разработки программных систем**

**Название лабораторной работы: Оценка эффективности и качества программы**

Студент гр. ИУ6-44  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шумаков А.А.**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Хорунжина К.С.**

(Подпись, дата) (И.О. Фамилия)

Москва, 2018

**Цель работы** - изучить известные критерии оценки и способы повышения эффективности и качества программных продуктов.

**Задание**

1. Исследовать заданный пример программы с целью определения ее эффективности и качества.
2. Определить основные критерии оценки и количественные характеристики для заданной программы.
3. Предложить варианты повышения эффективности и улучшения качества для заданного примера программы.

***Условие задачи***

Создать программу, в которой создается массив целых чисел в диапазоне от -20 до 5 и сортируется по убыванию (программа - v8. dpr).

**Исходный код программы:**

*{Обменная сортировка}*

*program V8;*

*{$APPTYPE CONSOLE}*

*uses*

*SysUtils;*

*const N=10;*

*var*

*m:array [1..N] of integer;*

*i,j,k,b,min:integer;*

*Begin*

*randomize;*

*for i:=1 to N do begin m[i]:=random(15)-20;write(' ',m[i]) end;*

*for j:=1 to N do*

*begin min:= m[j]; k:=j;*

*for i:=j to N do if m[i]<min then*

*begin min:=m[i]; k:=i end;*

*b:=m[j]; m[j]:=min; m[k]:=b;*

*end;*

*writeln;*

*for i:=1 to N do begin write(' ',m[i]) end;*

*readln;*

*end.*

В приведенной выше программе содержатся ошибки. Во-первых, массив m заполняется неверными числами из-за неверного аргумента функции random во время присваивания элемента массива. Так, в данном варианте программы массив будет заполнен в диапазоне [-20, -5), что противоречит условию задачи. Во-вторых, данная программа хоть и запустится, но результат ее выполнения будет неправильным, так как сортировка производится по возрастанию, а не по убыванию. В-третьих, при выводе массива в данном случаи тело цикла не нужно ограничивать операторными скобками begin…end, так как тело цикла содержит одну выполняемую операцию (вывод элемента сортированного массива).

Перепишем и улучшим данную программу, чтобы результат ее выполнения был правильным.

***Улучшения:***

1. Переход к более универсальной программы. Программа может работать с любым размером за счет использования динамического массива (индексация массива начинается с нуля);
2. Программа дает пользователю возможность ввода собственных чисел в массив, причем сразу же происходит проверка на принадлежность числа диапазону [-20,5), то есть проверка на ввод;
3. Переменная min замена на max, что облегчает понимание программы и более соответствует контексту данной задачи по сравнению исходного варианта;
4. Во внешнем и внутреннем циклах сортировки обмена уменьшено количество итераций, что привело к уменьшению времени работы программы;

**Код улучшенной программы:**

*{Обменная сортировка}*

*program V8new;*

*{$APPTYPE CONSOLE}*

*uses*

*SysUtils;*

*type Mymas=array of Integer;*

*var*

*mas:Mymas; // динамический массив*

*flag:boolean; // Флаг*

*i,j,k,b:integer; // min заменяем на max, чтобы имя переменной*

*max:Integer; // соответствовало контектсу задачи*

*Begin*

*j:=0;*

*flag:=False; //Флаг на правильность ввода размера массива*

*while (flag<>true) do*

*begin*

*Writeln('Enter size massive');*

*Readln(j);*

*if j>0 then flag:=True //Размер массива должен быть > 0*

*else Writeln('size massive should be >0');*

*end;*

*SetLength(mas,j);*

*i:=0;*

*flag:=false; //Флаг на полное заполнение массива*

*while (flag<>True) and (i<>Length(mas)) do*

*begin*

*Writeln('Enter number from -20 to 5');*

*Readln(b);*

*if (b>=-20) and (b<5) then // Проверка на требуемый диапазон*

*begin*

*mas[i]:=b;*

*Inc(i);*

*end*

*else*

*Writeln('Error Enter');*

*end;*

*Writeln('Massive:');*

*for j:=0 to Length(mas)-1 do*

*write(' ',mas[j]);*

*Writeln;*

*for j:=0 to Length(mas)-2 do // улучшили проход сократив число итераций*

*begin*

*max:= mas[j]; k:=j;*

*for i:=j+1 to Length(mas)-1 do if mas[i]>max then //итераций стало на одну //меньше*

*begin max:=mas[i]; k:=i end;*

*b:=mas[j]; mas[j]:=max; mas[k]:=b;*

*end;*

*writeln('Sort Massive:');*

*for i:=0 to Length(mas)-1 do*

*write(' ',mas[i]);*

*mas:=nil;*

*readln;*

*end.*

***Оценочные таблицы***

Тесты времени выполнения и памяти были проведены на тестовых данных, указанных в таблице 1 при размере массива равного 10. Результаты замера времени представлены в таблице 2. Для замера времени в исходной программе была исправлена часть кода так, чтобы программа выдавала правильный результат и имела возможность ввода с клавиатуры.

Оценка оперативной памяти выполнялась посредством ручного подсчета используемой памяти. Время считалось при помощи таймера: замерялось время начало сортировки, затем фиксировалось время окончания сортировки, потом вычли из времени окончания время начала и получили время работы программа, то есть:

, где

время начала сортировки;

время окончания сортировки;

время работы сортировки.

Таблица 1: Тесты для замера времени

|  |  |
| --- | --- |
| № теста | Входные данные для улучшенной программы |
| 1 | {4,3,2,1,0,-1,-2,-3,-4,-5} |
| 2 | {-5,-4,-3,-2,-1,0,1,2,3,4} |
| 3 | {1,2,3,4,0,1,2,3,4,0} |
| 4 | {-20,4,-19,3,-18,2,-17,1,-16,0} |
| 5 | {-20,-18,1,4,-1,-5,0,2,-8,3} |

Таблица 2: Результаты замеров времени для данных тестов

|  |  |  |
| --- | --- | --- |
| № теста | Время выполнения исходной программы, мс | Время выполнения улучшенной программы, мс |
| 1 | 936 | 558 |
| 2 | 1242 | 1107 |
| 3 | 909 | 597 |
| 4 | 943 | 868 |
| 5 | 1060 | 969 |
|  | 5090 | 4099 |
| Среднее время выполнения | 1018 | 820 |

Далее произведем оценку эффективности (таблица 3) и качества (таблицы 4,5) исходной и улучшенной программы.

*.*

Таблица 3: Оценка эффективности

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Исходная программа** | |  | **Улучшенная программа** | |
| **Критерий оценки** | **Недостатки** | **Колич-я**  **оценка** | **Улучшения** | | **Колич-я**  **оценка** |
| Время выполнения | Лишние итерации внешнего и внутреннего циклов при сортировки | 1018 мс | Нет лишних итераций  внешнего и внутреннего циклов при сортировки | | 820 мс |
| Оперативная память | Особых недостатков нет | 25 байт | Статический массив, размер которого задавался через константу N, заменен на динамический. Добавлена очистка памяти. | | 25 байт |
| Внешняя память | Не используется | - | Не используется | | - |

Таблица 4: Оценка качества исходной программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Правильность** | **Универсальность** | **Проверяемость** | **Точность**  **результатов** |
| Недостатки | Массив заполняется случайными числами из неверного диапазона, сортировка производится по возрастанию, а не по убыванию как указано в задании | Пользователь не может ввести размер массива N | Результаты программы выводятся на экран | Требование точности для данной программы не актуально |
| Оценка | - | - | + | + |

Таблица 5: Оценка качества улчшенной программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Правильность** | **Универсальность** | **Проверяемость** | **Точность**  **результатов** |
| Достоинства | Массив вводит пользователь при этом сразу же осуществляется проверка числа на принадлежность диапазону, указанному в задании | Размер массива задается пользователем  Обеспечен контроль ввода. | Все результаты выводятся на экран. Можно легко проверить их правильность. | Требование точности для данной программы не актуально |
| Оценка | + | +/+ | + | + |

**Вывод.** Изучены и применены на практике основные методы улучшения программных продуктов и критерии оценки их качества и эффективности.