МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Качество и метрология программного обеспечения»

ТЕМА: «Измерение характеристик динамической сложности программ с помощью профилировщика SAMPLER»

Студент гр. 6304 _____ Некрасов Н.А.

Преподаватель _____ Кирьянчиков В.А.

Санкт-Петербург

2020

Задание

1. Ознакомиться с документацией на монитор SAMPLER и выполнить под его управлением тестовые программы test_cyc.c и test_sub.c с анализом параметров повторения циклов, структуры описания циклов, способов профилирования процедур и проверкой их влияния на точность и чувствительность профилирования.

1. Скомпилировать и выполнить под управлением SAMPLER'a программу на С, разработанную в 1-ой лабораторной работе.

Выполнить разбиение программы на функциональные участки и снять профили для двух режимов:

1.- измерение только полного времени выполнения программы;

2.- измерение времен выполнения функциональных участков (ФУ).

Убедиться, что сумма времен выполнения ФУ соответствует полному времени выполнения программы.

3. Выявить "узкие места", связанные с ухудшением производительности программы, ввести в программу усовершенствования и получить новые профили. Объяснить смысл введенных модификаций программ.

Ход работы

Использовался старый SAMPLER. Программы компилировались с помощью Borland C++. Компилирование выполнялось на Ubuntu с использованием Wine и DosBox.

Тестовые программы

Код программы test_cyc.c с нумерацией строк представлен в приложении А.

Результаты профилирования:

```
-------------------------------------------------------------------
  NN          И м я  о б р а б о т а н н о г о  ф а й л а
-------------------------------------------------------------------
  1. TEST_CYC.CPP
-------------------------------------------------------------------
```

```
   Таблица с результатами измерений (
   используется 13 из 416 записей )

-----------------------------------------------------------------------
Исх.Поз. Прием.Поз.       Общее время(мкс)    Кол-во
прох. Среднее время(мкс)
-----------------------------------------------------------------------  1 : 8
                  1 :             10   4335.47     1     4335.47
-----------------------------------------------------------------------
 1 :  10  1 :  12            8675.98            1           8675.98
-----------------------------------------------------------------------
 1 :  12  1 :  14           21671.50            1          21671.50
-----------------------------------------------------------------------
 1 :  14  1 :  16           43348.87            1          43348.87
-----------------------------------------------------------------------
 1 :  16  1 :  19            4337.15            1           4337.15
-----------------------------------------------------------------------
 1 :  19  1 :  22            8668.43            1           8668.43
-----------------------------------------------------------------------
 1 :  22  1 :  25           21672.34            1          21672.34
-----------------------------------------------------------------------
 1 :  25  1 :  28           43348.03            1          43348.03
-----------------------------------------------------------------------
 1 :  28  1 :  34            4334.64            1           4334.64
-----------------------------------------------------------------------
 1 :  34  1 :  40            8670.11            1           8670.11
-----------------------------------------------------------------------
 1 :  40  1 :  46           21676.53            1          21676.53
-----------------------------------------------------------------------
 1 :  46  1 :  52           43348.87            1          43348.87
```

По результатам видно, что времена сильно завышены из-за накладных затрат эмулятора. В коде используется разная запись циклов с одинаковым количеством итераций, при этом отсутствует влияние на время. А также видна линейная зависимость времени от количества итераций.

Код программы test_sub.c с нумерацией строк представлен в приложении Б.

Результаты профилирования:

```
-----------------------------------------------------------------------
  NN          Имя обработанного файла
-----------------------------------------------------------------------
  1. TEST_SUB.CPP
-----------------------------------------------------------------------

   Таблица с результатами измерений (
   используется 5 из 416 записей )

-----------------------------------------------------------------------
Исх.Поз. Прием.Поз.       Общее время(мкс)    Кол-во
прох. Среднее время(мкс)
-----------------------------------------------------------------------
 1 :  30  1 :  32          433699.86            1         433699.86
-----------------------------------------------------------------------
 1 :  32  1 :  34          867392.18            1         867392.18
-----------------------------------------------------------------------
```

```
 1 :  34  1 :  36         2168480.87            1         2168480.87
-------------------------------------------------------------------
 1 :  36  1 :  38         4336949.16            1         4336949.16
-------------------------------------------------------------------
```

По результатам можно сделать аналогичные выводы о том, что время выполнения линейно зависит от количества итераций цикла и завышено ввиду использования эмулятора

Программа из ЛР1.

Код программы из первой лабораторной работы с нумерацией строк представлен в приложениях В (для измерения полного времени) и Г (для измерения времен выполнения ФУ).

Результаты профилирования с измерением полного времени:

```
-------------------------------------------------------------------
         Исх.Поз. Прием.Поз.  Общее время(мкс)
               Кол-во прох. Среднее время(мкс)
-------------------------------------------------------------------
 1 :  35  1 :  37           78255.59            1          78255.59
-------------------------------------------------------------------
```

Общее время выполнения первой функции – 78255 мкс. Результаты завышены из-за затрат на работу эмулятора, а также тем, что ноутбук был дважды залит водой.

Результаты профилирования с измерением времен ФУ:

```
  Таблица с результатами измерений (
 используется 8 из 416 записей )

-------------------------------------------------------------------
Исх.Поз. Прием.Поз.  Общее время(мкс)
 Кол-во прох. Среднее время(мкс)
-------------------------------------------------------------------
 1 :  18  1 :  20              0.00         4858              0.00
-------------------------------------------------------------------
 1 :  20  1 :  22          80163.93         4858             16.50
-------------------------------------------------------------------
 1 :  22  1 :  29       83727430.90        37549           2229.82
 1 :  22  1 :  25         126320.42        34018              3.71
-------------------------------------------------------------------
 1 :  25  1 :  27          74247.81        34018              2.18
-------------------------------------------------------------------
 1 :  27  1 :  29         207960.24        34018              6.11
-------------------------------------------------------------------
 1 :  29  1 :  22       81255254.87         1173          69271.32
 1 :  29  1 :  31              0.00         4858              0.00
-------------------------------------------------------------------
```

```
 1 :  31  1 :  33           26135.20         4858              5.38
-----------------------------------------------------------------------
 1 :  33  1 :  18               0.00         4857              0.00
-----------------------------------------------------------------------
```

Время измерения на функциональном уровне сильно завышено, либо является нулевым. Предположительно причиной этому является запуск на виртуальной машине, а также особенности профилировщика

Измененная программа из первой лабораторной работы

Измененный код программы из первой лабораторной работы с нумерацией строк представлен в приложениях Д (для измерения полного времени).

Результаты профилирования с измерением полного времени:

```
-----------------------------------------------------------------------
         Исх.Поз. Прием.Поз. Общее время(мкс)
               Кол-во прох. Среднее время(мкс)
-----------------------------------------------------------------------
 1 :  30  1 :  32           9135.25           1           9135.25
-----------------------------------------------------------------------
```

Общее время выполнения программы уменьшилось на 89% и составило 9135.2 мкс. Причиной такого сильного сокращения времени заключается во вставке тела функции swap в тело selectionSort, ввиду чего не потребовалось тратить время на вызов функции.

Результаты профилирования с измерением времен ФУ (код в приложении Е):

```
-----------------------------------------------------------------------
Исх.Поз. Прием.Поз. Общее время(мкс) Кол-во прох.
Среднее время(мкс)
-----------------------------------------------------------------------
 1 :  11  1 :  14            3287.85         4858              0.68
-----------------------------------------------------------------------
 1 :  14  1 :  19 15460151924864239600.00         4858   3182410853203837.00
-----------------------------------------------------------------------
```

```
 1 :  19  1 :  21           86339.86          4858              17.77
-------------------------------------------------------------------------
 1 :  21  1 :  25            4279.32          4858               0.88
-------------------------------------------------------------------------
 1 :  25  1 :  11            3028.04          4857               0.62
-------------------------------------------------------------------------
```

По результатам измерений времени на ФУ видно, что время выполнения участка, отвечающего за обмен значений переменных и правда уменьшился, что подтверждает выдвинутое ранее предположение.

Выводы

В результате выполнения данной лабораторной работы был изучен монитор SAMPLER, с помощью которого было выполнено профилирование тестовых программ test_cyc.c и test_sub.c.

Было проанализировано полное время выполнения программы, разработанной в 1-ой лабораторной работе, и время выполнения её ФУ.

ПРИЛОЖЕНИЕ А

TEST_CYC.C

```
1. #define Size 10000
2. int i, tmp, dim[Size];
3.
4. void main()
5. {
6.      for(i=0;i<Size/10;i++){ tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
7.      for(i=0;i<Size/5;i++){ tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
8.      for(i=0;i<Size/2;i++){ tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
9.      for(i=0;i<Size;i++)  { tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
10.        for(i=0;i<Size/10;i++)
11.          { tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
12.        for(i=0;i<Size/5;i++)
13.          { tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
14.        for(i=0;i<Size/2;i++)
15.          { tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
16.        for(i=0;i<Size;i++)
17.          { tmp=dim[0]; dim[0]=dim[i]; dim[i]=tmp; };
18.        for(i=0;i<Size/10;i++)
19.          { tmp=dim[0];
20.            dim[0]=dim[i];
21.            dim[i]=tmp;
22.          };
23.        for(i=0;i<Size/5;i++)
24.          { tmp=dim[0];
25.            dim[0]=dim[i];
26.            dim[i]=tmp;
27.          };
28.        for(i=0;i<Size/2;i++)
29.          { tmp=dim[0];
30.            dim[0]=dim[i];
31.            dim[i]=tmp;
32.          };
33.        for(i=0;i<Size;i++)
34.          { tmp=dim[0];
35.            dim[0]=dim[i];
36.            dim[i]=tmp;
37.          };
38.   }
```

ПРИЛОЖЕНИЕ Б

TEST_SUB.C

```
1. const unsigned Size = 1000;
2.
3.
4. void TestLoop(int nTimes)
5. {
6.   static int TestDim[Size];
7.   int tmp;
8.   int iLoop;
9.
10.  while (nTimes > 0)
11.  {
12.    nTimes --;
13.
14.    iLoop = Size;
15.    while (iLoop > 0)
16.    {
17.      iLoop -- ;
18.      tmp = TestDim[0];
19.      TestDim[0] = TestDim[nTimes];
20.      TestDim[nTimes] = tmp;
21.    }
22.  }
23.} /* TestLoop */
24.
25.
26.void main()
27.{
28.   TestLoop(Size / 10); // 100 * 1000  п о в т о р е н и й
29.   TestLoop(Size / 5);  // 200 * 1000  п о в т о р е н и й
30.   TestLoop(Size / 2);  // 500 * 1000  п о в т о р е н и й
31.   TestLoop(Size / 1);  // 1000* 1000  п о в т о р е н и й
32.}
```

ПРИЛОЖЕНИЕ В

Полное время default.cpp

```
1. #include <stdio.h>
2. #include <time.h>
3. #include <stdlib.h>
4.
5. #include "Sampler.h"
6.
7. #define MAX 4859
8.
9. void swap(int* a, int* b)
10. {
11. int tmp = *a;
12. *a = *b;
13. *b = tmp;
14. }
15.
16. void selectionSort(int array[], int size) {
17. for (int i = 0; i < size - 1; i++) {
18. int maxIndex = i;
19. for (int j = i + 1; j < size; j++) {
20. if (array[j] > array[maxIndex])
21. maxIndex = j;
22. }
23. swap(&array[i], &array[maxIndex]);
24. }
25. }
26.
27. int main()
28. {
29. srand(time(NULL));
30. int array[MAX];
31.
32. for(int i = 0; i < MAX; ++i)
33. array[i] = rand();
34.
35. SAMPLE;
36. selectionSort(array, MAX);
37. SAMPLE;
38.
39. return 0;
40. }
```

ПРИЛОЖЕНИЕ Г

Функциональные замеры def_func.c

```
1. #include <stdio.h>
2. #include <time.h>
3. #include <stdlib.h>
4.
5. #include "Sampler.h"
6.
7. #define MAX 4859
8.
9. void swap(int* a, int* b)
10. {
11. int tmp = *a;
12. *a = *b;
13. *b = tmp;
14. }
15.
16. void selectionSort(int array[], int size) {
17. for (int i = 0; i < size - 1; i++) {
18. SAMPLE;
19. int maxIndex = i;
20. SAMPLE;
21. for (int j = i + 1; j < size; j++) {
22. SAMPLE;
23. if (array[j] > array[maxIndex])
24. {
25. SAMPLE;
26. maxIndex = j;
27. SAMPLE;
28. }
29. SAMPLE;
30. }
31. SAMPLE;
32. swap(&array[i], &array[maxIndex]);
33. SAMPLE;
34. }
35. }
36.
37. int main()
38. {
39. srand(999);
40. int array[MAX];
41.
42. for(int i = 0; i < MAX; ++i)
43. array[i] = rand();
44.
45. selectionSort(array, MAX);
46.
47. return 0;
48. }
```

ПРИЛОЖЕНИЕ Д

Модифицированный код mod.cpp

```
1. #include <stdio.h>
2. #include <time.h>
3. #include <stdlib.h>
4.
5. #include "Sampler.h"
6.
7. #define MAX 4859
8.
9. void selectionSort(int array[], int size) {
10. for (int i = 0; i < size - 1; i++) {
11. int maxIndex = i;
12. for (int j = i + 1; j < size; j++) {
13. if (array[j] > array[maxIndex])
14. maxIndex = j;
15. }
16. int tmp = array[i];
17. array[i] = array[maxIndex];
18. array[maxIndex] = tmp;
19. }
20. }
21.
22. int main()
23. {
24. srand(132);
25. int array[MAX];
26.
27. for(int i = 0; i < MAX; ++i)
28. array[i] = rand();
29.
30. SAMPLE;
31. selectionSort(array, MAX);
32. SAMPLE;
33.
34. return 0;
35. }
```

ПРИЛОЖЕНИЕ Е

Модифицированный код функциональных измерений mod_func.cpp

```
1. #include <stdio.h>
2. #include <time.h>
3. #include <stdlib.h>
4.
5. #include "Sampler.h"
6.
7. #define MAX 4859
8.
9. void selectionSort(int array[], int size) {
10. for (int i = 0; i < size - 1; i++) {
11. SAMPLE;
12. int maxIndex = i;
13.
14. SAMPLE;
15. for (int j = i + 1; j < size; j++) {
16. if (array[j] > array[maxIndex])
17. maxIndex = j;
18. }
19. SAMPLE;
20.
21. SAMPLE;
22. int tmp = array[i];
23. array[i] = array[maxIndex];
24. array[maxIndex] = tmp;
25. SAMPLE;
26. }
27. }
28.
29. int main()
30. {
31. srand(132);
32. int array[MAX];
33.
34. for(int i = 0; i < MAX; ++i)
35. array[i] = rand();
36.
37. selectionSort(array, MAX);
38.
39. return 0;
40. }
```