МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МОЭВМ

ОТЧЕТ

по практической работе №3

по дисциплине «Качество и метрология программного обеспечения»

ТЕМА: ИЗМЕРЕНИЕ ХАРАКТЕРИСТИК ДИНАМИЧЕСКОЙ СЛОЖНОСТИ ПРОГРАММ С ПОМОЩЬЮ ПРОФИЛИРОВЩИКА SAMPLER_v2

Студент гр. 8304 Нам Ё Себ

Преподаватель Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить возможности измерения динамических характеристик программ с помощью профилировщика на примере профилировщика SAMPLER

Ход работы.

1. Были выполнены тестовые программы test_cyc.c и test_sub.c под управлением SAMPLER

Таблица 1 – Результаты для test_cyc.c

исх	прием	общее время	кол-во проходов	среднее время
13	15	5810.000	1	5810.000
15	17	9190.000	1	9190.000
17	19	23010.000	1	23010.000
19	21	38340.000	1	38340.000
21	24	2890.000	1	2890.000
24	27	5690.000	1	5690.000
27	30	14210.000	1	14210.000
30	33	28260.000	1	28260.000
33	39	2890.000	1	2890.000
39	45	5670.000	1	5670.000
45	51	14150.000	1	14150.000
51	57	28250.000	1	28250.000

Таблица 2 – Результаты для test_sub.c

исх	прием	общее время	кол-во проходов	среднее время
30	32	25913830.000	1	25913830.000
32	34	51927060.000	1	51927060.000
34	36	131363580.000	1	131363580.000
36	38	262845350.000	1	262845350.000

2. Выполнили программу из ЛР1 под управлением Sampler с внешним зацикливанием и получили отчет по результатам профилирования.

исх	прием	общее время	кол-во проходов	среднее время
91	21	10.000	1	10.000
21	41	170.000	1	170.000
41	43	680.000	57	11.930
43	47	6550.000	130	50.385
47	49	590.000	130	4.385
49	53	3110.000	130	23.923
53	43	5740.000	73	78.630
53	58	5590.000	57	98.070
58	41	3700.000	56	66.071
58	77	50.000	1	50.000
77	93	40.000	1	40.000

3. Выполнили оптимизацию на участке с 47 по 49 строку. В результате получили следующее:

исх	прием	общее время	кол-во проходов	среднее время
93	22	-30.000	1	-30.000
22	42	130.000	1	130.000
42	44	660.000	52	12.692
44	48	3910.000	146	26.781
48	50	580.000	146	3.973
50	54	2580.000	146	17.671
54	44	1880.000	94	20.000
54	59	1200.000	52	23.077
59	42	3940.000	51	77.255
59	78	100.000	1	100.000
78	95	20.000	1	20.000

Оптимизация данного участка заключалась в замене постфиксного инкремента переменной j на префиксный. Данная замена ускоряет выполнение программы так как при выполнении декремента создается копия переменной в отличие от инкремента.

Заключение

В ходе лабораторной работы изучили возможности измерения динамических характеристик программ с помощью профилировщика на примере профилировщика SAMPLER.

Приложение А

Исходный код программы

```
// Online C compiler to run C program online
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
int max = 80;

void swap(int *p, int *q){
    int hold;
    hold = *p;
    *p = *q;
    *q = hold;
}

void sort(int x[max], int n){
    int left[20], right[20], i, j, sp, mid;
    float pivot;
    left[0] = 0;
    right[0] = n - 1;
    sp = 0;
SAMPLE;
    while (sp > -1){
        if (left[sp] >= right[sp]){
            sp--;
        }
        else {
            i = left[sp];
            j = right[sp];
            pivot = x[j];
            mid = (i + j) / 2;
            if ((j - i) > 5){
                if (((x[mid] < pivot) && (x[mid] > x[i])) ||
((x[mid] > pivot) && (x[mid] < x[i]))){
                    swap(&(x[mid]), &(x[j]));
                }
                else if(((x[i] < x[mid]) && (x[i] > pivot)) ||
((x[i] > x[mid]) && (x[i] < pivot))){
                    swap(&(x[i]), &(x[j]));
                }
            }

            pivot = x[j];
SAMPLE;
        while (i < j){
```

```
SAMPLE;
            while (x[i] < pivot){
                i++;
            }
            SAMPLE;
            j--;
            SAMPLE;
            while ((i < j) && (pivot < x[j])){
                j--;
            }
            SAMPLE;
            if (i < j){
                    swap(&(x[i]), &(x[j]));
                }
        }
        SAMPLE;
        j = right[sp];

        swap(&(x[i]), &(x[j]));

        if (i - left[sp] >= right[sp] - i){
            left[sp + 1] = left[sp];
            right[sp + 1] = i - 1;
            left[sp] = i + 1;
        }
        else{
        left[sp + 1] = i + 1;
        right[sp + 1] = right[sp];
        right[sp] = i - 1 ;
        }

        sp++;
        }
    }
SAMPLE;
}


int main() {
    int n = max;
    int x[n];

    srand(time(NULL));

    for (int i = 0; i < n; i++){
```

```
    x[i] = rand()%100;
    }
    SAMPLE;
    sort(x, n);
    SAMPLE;
    for(int i = 0; i < n; i++){
        printf("%d ", x[i]);
    }

    return 0;
}
```

Приложение В
Код программы после оптимизации

```
// Online C compiler to run C program online
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include "sampler.h"

int max = 80;

void swap(int *p, int *q){
    int hold;
    hold = *p;
    *p = *q;
    *q = hold;
}

void sort(int x[max], int n){
    int left[20], right[20], i, j, sp, mid;
    float pivot;
    left[0] = 0;
    right[0] = n - 1;
    sp = 0;
SAMPLE;
    while (sp > -1){
        if (left[sp] >= right[sp]){
            --sp;
        }
        else {
            i = left[sp];
            j = right[sp];
            pivot = x[j];
            mid = (i + j) / 2;
```

```
            if ((j - i) > 5){
                if (((x[mid] < pivot) && (x[mid] > x[i])) ||
((x[mid] > pivot) && (x[mid] < x[i]))){
                    swap(&(x[mid]), &(x[j]));
                }
                else if(((x[i] < x[mid]) && (x[i] > pivot)) ||
((x[i] > x[mid]) && (x[i] < pivot))){
                    swap(&(x[i]), &(x[j]));
                }
            }

            pivot = x[j];
SAMPLE;
        while (i < j){
SAMPLE;
            while (x[i] < pivot){
                i++;
            }
            SAMPLE;
            --j;
            SAMPLE;
            while ((i < j) && (pivot < x[j])){
                j--;
            }
            SAMPLE;
            if (i < j){
                    swap(&(x[i]), &(x[j]));
                }
        }
        SAMPLE;
        j = right[sp];

        swap(&(x[i]), &(x[j]));

        if (i - left[sp] >= right[sp] - i){
            left[sp + 1] = left[sp];
            right[sp + 1] = i - 1;
            left[sp] = i + 1;
        }
        else{
        left[sp + 1] = i + 1;
        right[sp + 1] = right[sp];
        right[sp] = i - 1 ;
        }
```

```
        sp++;
        }
    }
SAMPLE;
}


int main(int pargc, char **argv) {
sampler_init(&pargc, argv);

    int n = max;
    int x[n];

    srand(time(NULL));

    for (int i = 0; i < n; i++){
    x[i] = rand()%100;
    }
    SAMPLE;
    sort(x, n);
    SAMPLE;
    //for(int i = 0; i < n; i++){
//      printf("%d ", x[i]);
    //}

    return 0;
}
```