آزمایش شماره ی 5 **آزمایشگاه سیستم عامل**

نام و نام خانوادگی: هلیا سادات هاشمی پور **شماره دانشجویی:**9831106

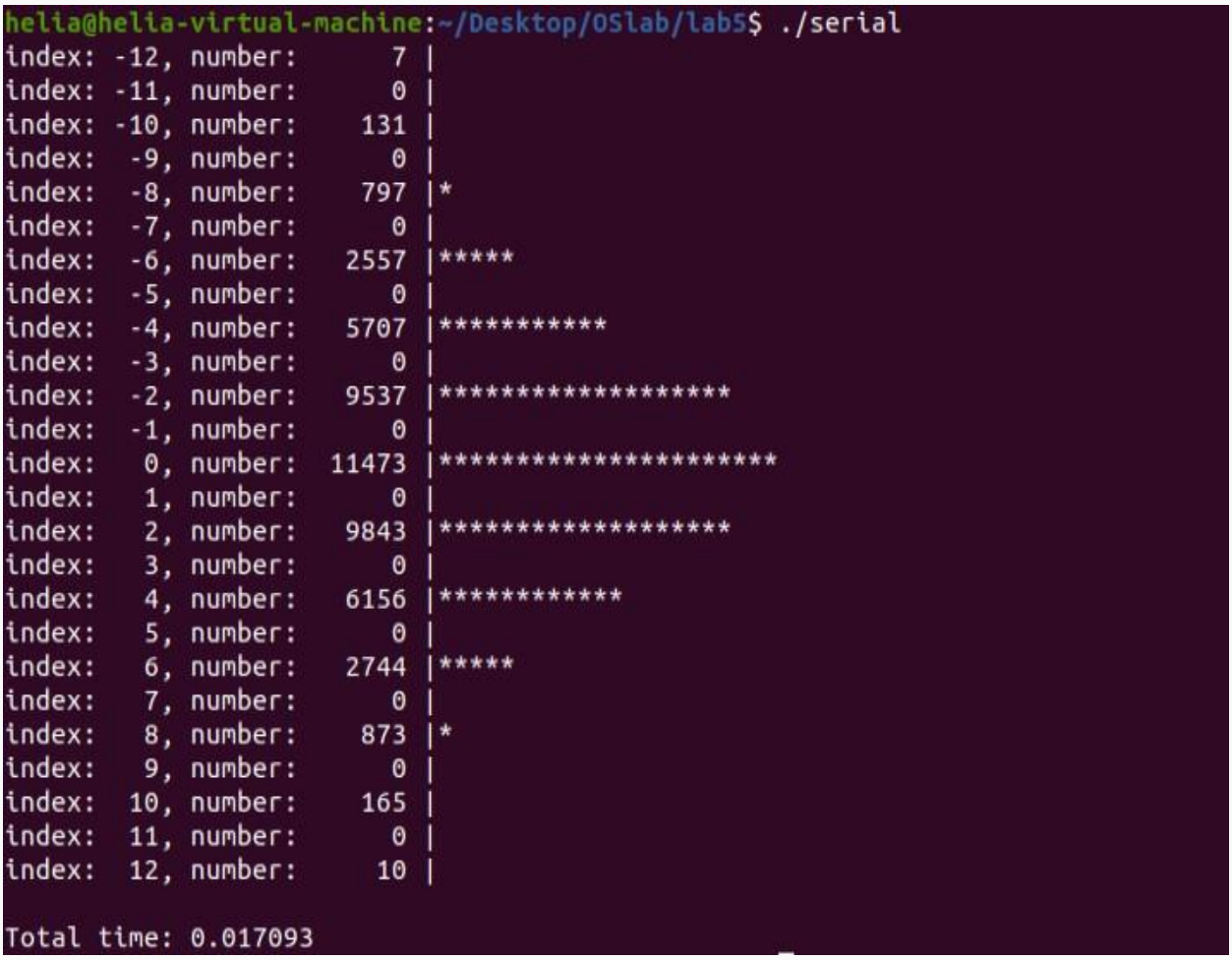
نام استاد:استاد علیزاده **تاریخ:**26 آبان 1400

-1

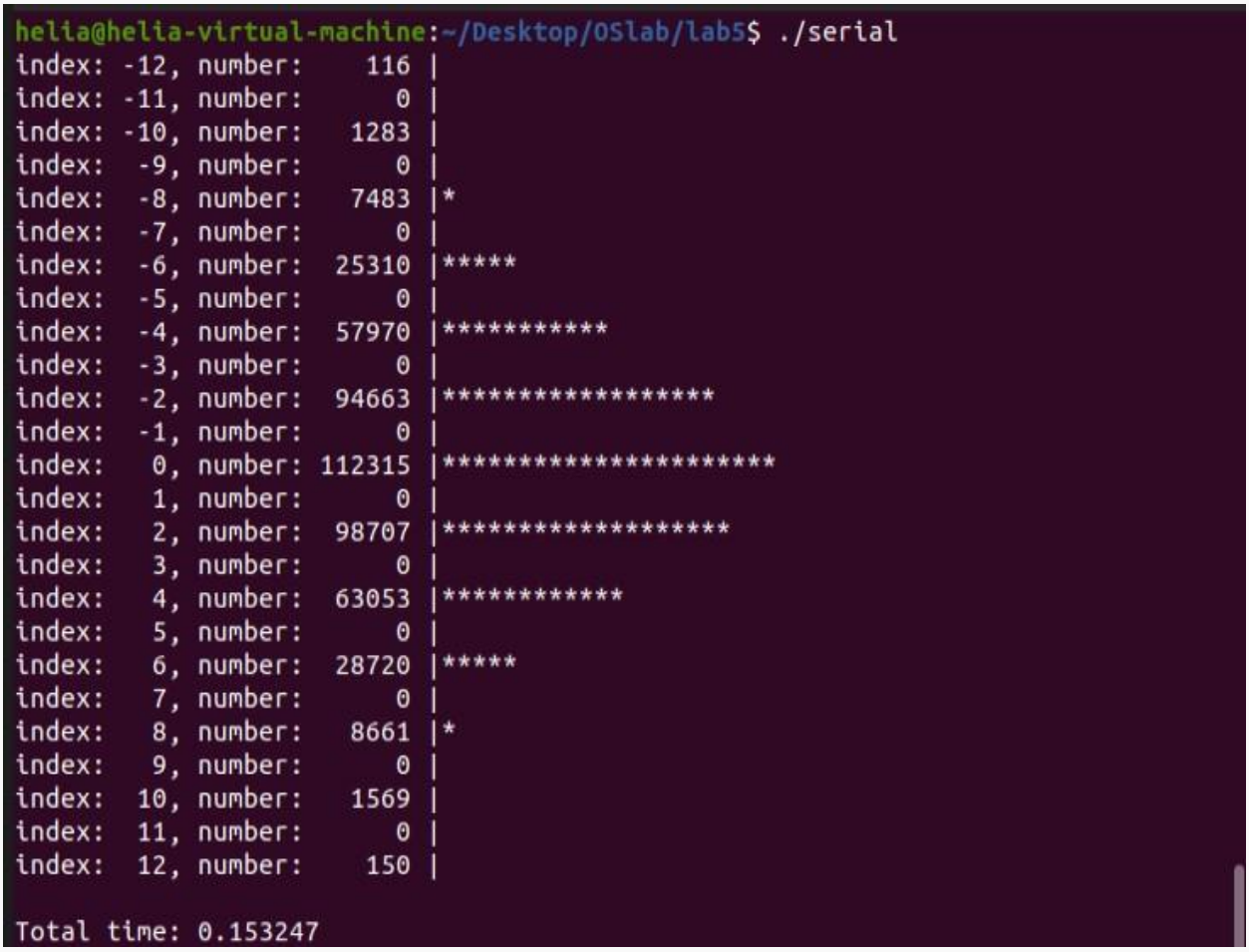
نمونه 5000

```
helia@helia-virtual-machine:~/Desktop/OSlab/lab5$ ./serial
index: -12, number:      1 |
index: -11, number:      0 |
index: -10, number:     11 |
index:  -9, number:      0 |
index:  -8, number:     69 |*
index:  -7, number:      0 |
index:  -6, number:    246 |****
index:  -5, number:      0 |
index:  -4, number:    561 |***********
index:  -3, number:      0 |
index:  -2, number:    960 |*******************
index:  -1, number:      0 |
index:   0, number:   1134 |**********************
index:   1, number:      0 |
index:   2, number:    976 |*******************
index:   3, number:      0 |
index:   4, number:    659 |*************
index:   5, number:      0 |
index:   6, number:    287 |*****
index:   7, number:      0 |
index:   8, number:     83 |*
index:   9, number:      0 |
index:  10, number:     13 |
index:  11, number:      0 |
index:  12, number:      0 |

Total time: 0.002007
```

نمونه 50000


helia@helia-virtual-machine:~/Desktop/OSlab/lab5$ ./serial
index: -12, number: 7 |
index: -11, number: 0 |
index: -10, number: 131 |
index: -9, number: 0 |
index: -8, number: 797 |*
index: -7, number: 0 |
index: -6, number: 2557 |*****
index: -5, number: 0 |
index: -4, number: 5707 |**********
index: -3, number: 0 |
index: -2, number: 9537 |******************
index: -1, number: 0 |
index: 0, number: 11473 |**********************
index: 1, number: 0 |
index: 2, number: 9843 |******************
index: 3, number: 0 |
index: 4, number: 6156 |***********
index: 5, number: 0 |
index: 6, number: 2744 |*****
index: 7, number: 0 |
index: 8, number: 873 |*
index: 9, number: 0 |
index: 10, number: 165 |
index: 11, number: 0 |
index: 12, number: 10 |
Total time: 0.017093

نمونه 50000



زمان اجرای هر کدام بر حسب ثانیه

تعداد نمونه	**5000**	**50000**	**500000**
زمان اجرا	0.002007	0.017093	0.153247

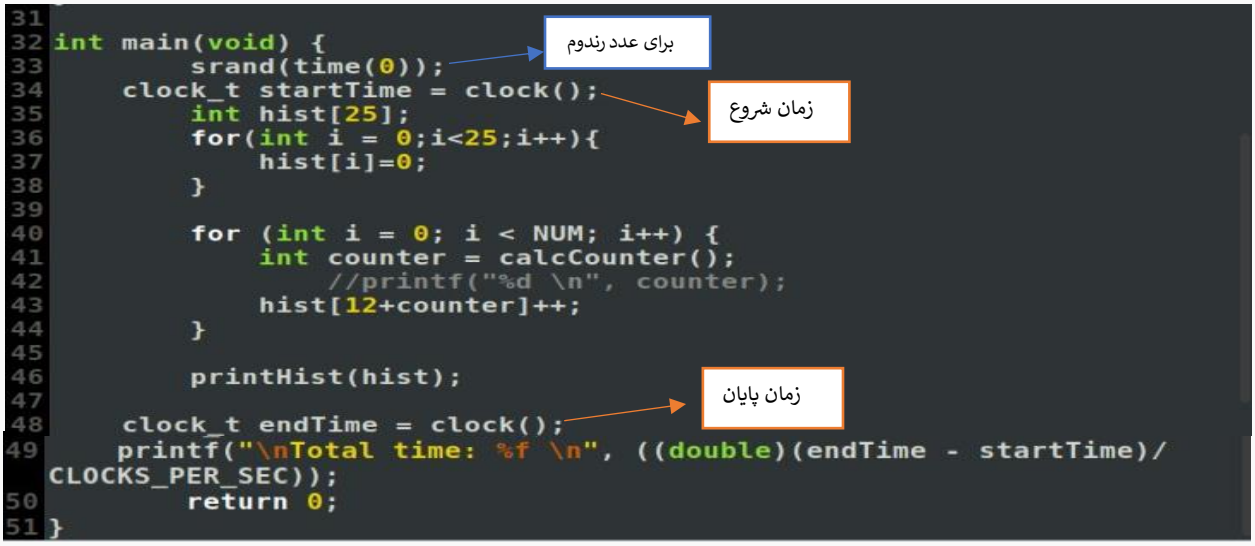
کد

تابع calcCounterبرای محاسبه 12 مقدار رندوم طبق توضیحات دستورکار است و مقدار را از می گرداند.تابع printHist مانند دستورکار برای چاپ نتایج نوشته شده است تا بتوانیم خروجی ر که به صورت نمودار منحنی نرمال خواهد بود را مشاهده بکنیم. برای اینکه نتایج خروجی ما در یک سطر قابل مشاهده باشد ،تعداد ستاره هایی که اید در هر ردیف چاپ بشود را بر مقداری که با * مشخص شده است تقسیم کرده ایم .

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define NUM 500000

int calcCounter(){
    int counter = 0;
        for(int i=0; i<12; i++){
            int random = rand()%101;
                //printf("%d ", random);
            if(random>49){
                counter++;
            }else{
                counter--;
            }
        }
        //printf("\n");
        return counter;
}

void printHist(int *hist){
    for (int i = 0; i < 25; i++) {
                printf("index: %3d, number: %6d |",i-12,hist[i]);
        for (int j = 0; j < hist[i]/(NUM/100); j++) {
            printf("*");
        }
        printf("\n");
    }
}
```

ابتدا آرایه را مقداردهی اولیه کرده سپس به تعداد نمونه های مورد نظر تابع را صدا زده و مقدار خانه های آرایه را تغییر می دهیم.

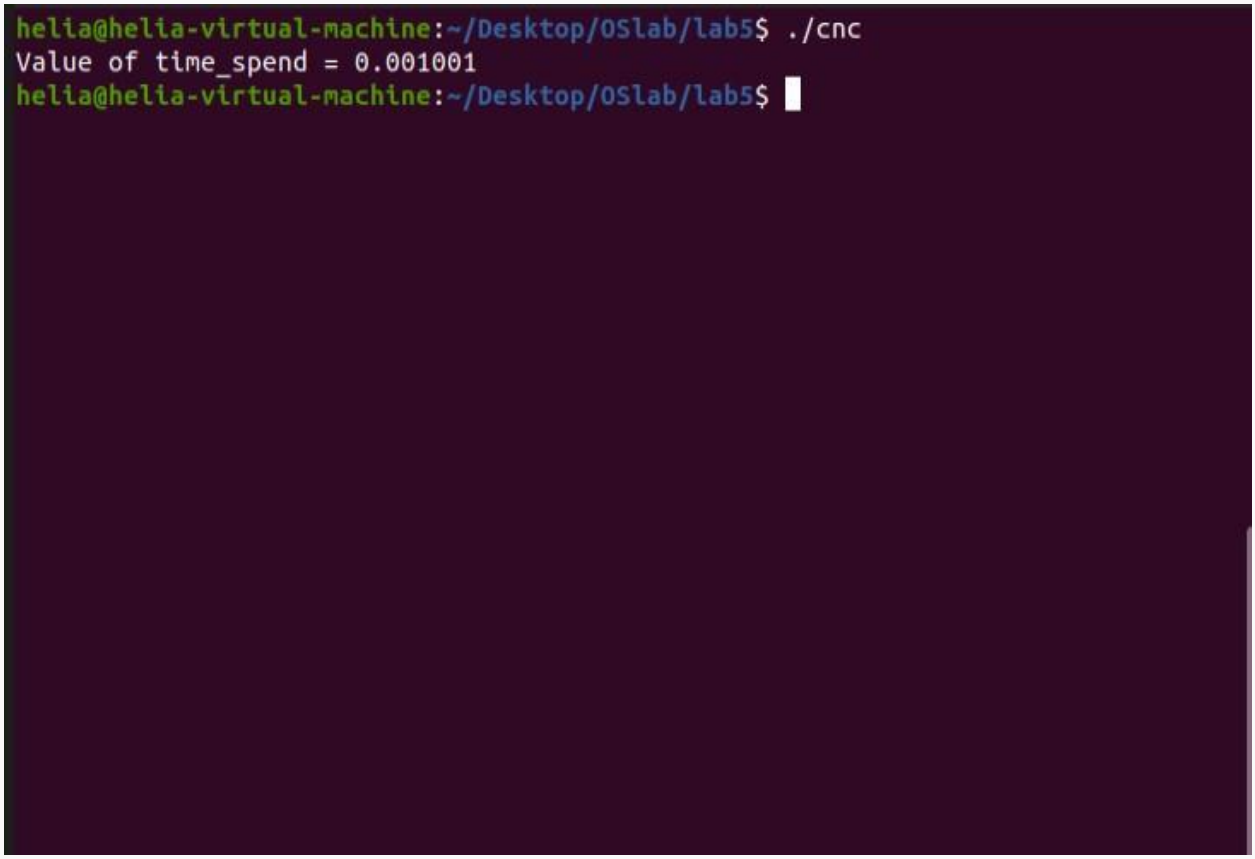
با صدا زدن تابع calcCounter خروجی ها را چاپ می کنیم و در انتها با به دست آوردن اختلاف زمان شروع و پایان و تقسیم آنها به CLOCK_PER_SECOND زمانی که اجرا طول می کشد را بر حسب ثانیه محاسبه می کند و در ترمینال چاپ می کند.



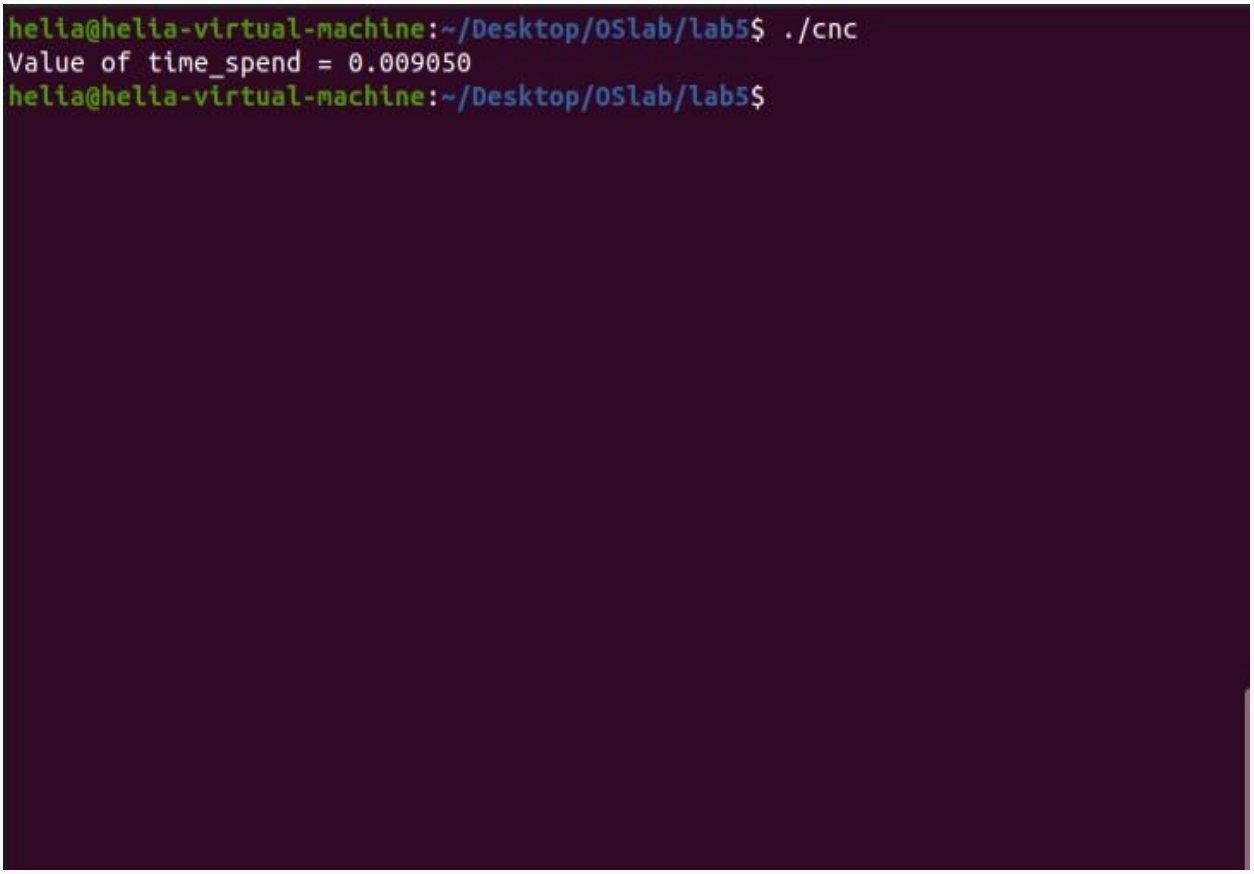

```
31
32 int main(void) {
33         srand(time(0));
34     clock_t startTime = clock();
35         int hist[25];
36         for(int i = 0;i<25;i++){
37             hist[i]=0;
38         }
39
40         for (int i = 0; i < NUM; i++) {
41             int counter = calcCounter();
42                 //printf("%d \n", counter);
43             hist[12+counter]++;
44         }
45
46         printHist(hist);
47
48     clock_t endTime = clock();
49     printf("\nTotal time: %f \n", ((double)(endTime - startTime)/
   CLOCKS_PER_SEC));
50         return 0;
51 }
```

2

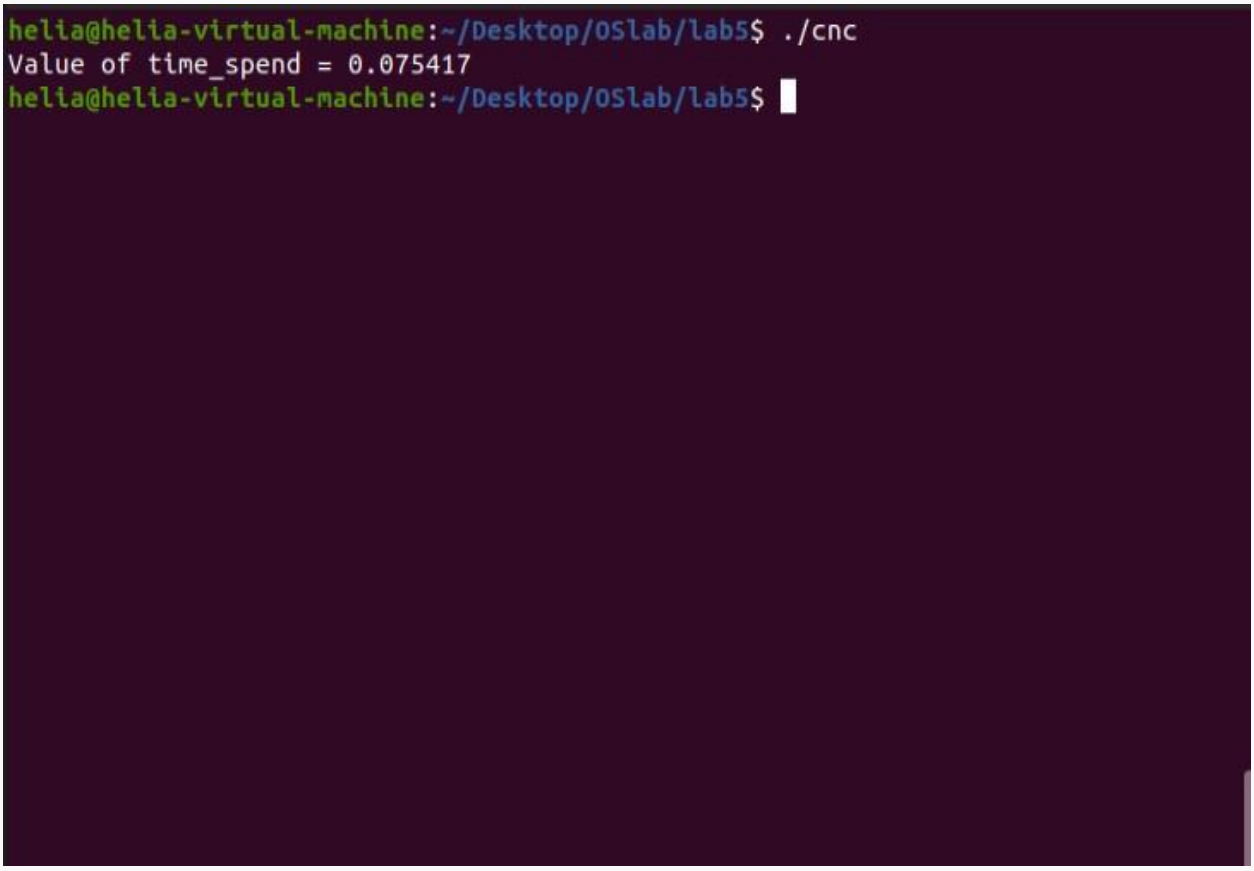
نمونه 5000


helia@helia-virtual-machine:~/Desktop/OSlab/lab5$ ./cnc
Value of time_spend = 0.001001
helia@helia-virtual-machine:~/Desktop/OSlab/lab5$

نمونه 50000


helia@helia-virtual-machine:~/Desktop/OSlab/lab5$ ./cnc
Value of time_spend = 0.009050
helia@helia-virtual-machine:~/Desktop/OSlab/lab5$

نمونه 500000



زمان اجرای هر کدام بر حسب ثانیه

تعداد نمونه	5000	50000	500000
زمان اجرا	0.001001	0.009050	0.075417

کد

در این بخش دو فرآیند والد و فرزند ایجاد کردم و از یک حافظه ی مشترک برایشان ایجاد کرده ام تا فرآیند والد و فرزند از آن جهت دسترسی به آرایه استفاده بکنند.

تابع shmget می خواهد آن حافظه را ایجاد بکند و پارامترهای مربوطه را به آن می دهیم.

Shamat هم برای attach مموری هست سپس هنگامی که کار با مموری تمام شد آن را detach می کند و برای اینکه به شکل کلی حافظه را از بین بریم دستور shmctl را داریم.

```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <semaphore.h>
#include <pthread.h>
#define NUM 50000
#define SHM_KEY 6868
#define SHM_SIZE sizeof(int) * 25

sem_t mutex;

int create_segment() {
    int shmid;
    key_t key;
    char *shm;

    key = SHM_KEY;

    if ((shmid = shmget(key, 1024, 0666 | IPC_CREAT)) < 0) {
        perror("shmget");
        exit(1);
    }
    return shmid;
}

int* attach_segment(int shmid) {
    int *shm;

    if ((shm = shmat(shmid, NULL, 0)) == (int *) -1) {
        perror("shmat");
        exit(1);
    }

    return shm;
}
void detach_segment(int *shm) {
    if (shmdt(shm) == -1) {
        perror("shmdt");
        exit(1);
    }
}
void remove_segment(int shmid) {
    if (shmctl(shmid, IPC_RMID, NULL) == -1) {
        perror("shmctl");
```

این قسمت هم همچون قسمت قبلی هست(برای چاپ hiostogram)

فقط با توجه به صورت دستور کار در خط 69 از fork استفاده کرده ام.و پردازه ی فرزند (pid==0)داخل if اجرا می شود و در آخر آن از دستور exit(0) استفاده کرده ایم. در صورتی که pid کوچک تر از صفر بباشد پیغام مناسب چاپ می شود و اگر این دو شرط نبود فرآیند والد اجرا می شود.

```
49          exit(1);
50      }
51  }
52
53  void print_histogram(int *hist, int number_of_samples) {
54      printf("Histogram for sample %d:\n", number_of_samples);
55      for (int i = 0; i < 25; i++) {
56          for (int j = 0; j < hist[i]; j++) {
57              printf("*");
58          }
59          printf("\n");
60      }
61  }
62
63  double calculate(int number_of_samples) {
64      clock_t begin = clock();
65      int shmid = create_segment();
66      int* hist = attach_segment(shmid);
67      srand(time(0));
68      int rand_num, counter;
69      int pid = fork();
70      if (pid < 0){
71                  printf("Cannot create child process");
72                  exit(-1);
73          }else if (pid == 0){
74          for (int i = 0; i < number_of_samples / 2; i++){
75              counter = 0;
76              for (int j = 0; j < 12; j++) {
77                  rand_num = rand() %101;
78                  if (rand_num >= 49)
79                      counter +=1;
80                  else
81                      counter -=1;
82              }
83              hist[counter +12] += 1;
84          }
85          exit(0);
86      }
87      else {
88          for (int i = 0; i < number_of_samples / 2; i++) {
89              counter = 0;
90              for (int j = 0; j < 12; j++) {
91                  rand_num = rand() % 101;
92                  if (rand_num >= 49)
93                      counter += 1;
94                  else
95                      counter -= 1;
96              }
```

زمانی CLOCK_PER_SECONDدر انتها با به دست آوردن اختلاف زمان شروع و پایان و تقسیم آنها به که اجرا طول می کشد را بر حسب ثانیه محاسبه می کند و در ترمینال چاپ می کند.

```
96              }
97              hist[counter + 12] ++;
98          }
99      }
100
101     //print_histogram(hist, number_of_samples);
102     detach_segment(hist);
103     remove_segment(shmid);
104
105     clock_t end = clock();
106     double time_spend = end - begin;
107     return time_spend;
108 }
109
110 int main() {
111     printf("\nTotal time: %f \n",((double)(calculate(NUM)/
    CLOCKS_PER_SEC)));
112
113     return 0;
114 }
```

پاسخ به سوالات

آیا این برنامه درگیر شرایط مسابقه می شود؟چگونه؟ اگر جوابتان مثبت بود راه حلی برای آن بیابید.

بله، در شرایطی که دو پردازه فرزند بخواهند مقدار یک خانه از آرایه data که در حافظه مشترک است را تغییر دهند ممکن است دچار شرایط مسابقه بشوند و مقدار آن خانه را به اشتباه تغییر دهند.که برای حل این مشکل از semaphore استفاده کرده ایم.(در اصل در اینجا متغیر hist ایجاد شرایط مسابقه را می کند که برای رفع آن می توان از spin lock و یا semaphore، انحصار متقابل در حین دسترسی به آرایه hist ایجاد کرد)

نتایج قسمت اول و دوم را مقایسه کنید و میزان افزایش سرعت را در جدول گزارش دهید.

تعداد نمونه	**5000**	**50000**	**500000**
اختلاف زمانها	**0.001006**	**0.008043**	**0.07783**

با توجه به جدول می بینیم که اختلاف زمان در قسمت دوم همواره بهتر از قسمت اول عمل هست در اصل کدیِ که نوشتیم باعث بهبود سرعت شده است.با داشتن فرایند والد و فرزند سرعت برنامه ما بهبود داشته است.(به دلیل اینکه از موازی سازی و همروندی در برنامه بیشتر استفاده شده است و کارها تقسیم شده اند.)