

به نام خدا

داکیومنت آزمایشگاه OS پایان ترم

نیوشا یقینی – 98522346

*پاسخ هر سوال در یک صفحه مجزا است.

1. در قالب یک دستور نام گروه های سیستم را از فایل `/etc/group` جدا کرده و در فایل جدیدی ذخیره کنید. سپس، برای این فایل دسترسی نوشتن و جستجو کردن را به اعضای گروه بدهید.

```
niusha@niusha-ygh:~$ cd Desktop/
niusha@niusha-ygh:~/Desktop$ cut -d ':' -f 1 /etc/group > grp_names.txt
niusha@niusha-ygh:~/Desktop$ chmod g+wx grp_names.txt
niusha@niusha-ygh:~/Desktop$ ls -l
total 32
-rwxrwxr-x 1 niusha niusha 16040 May 24 22:29 a.out
-rw-rwxr-- 1 niusha niusha 562 Jun 8 12:49 grp_names.txt
-rw-rw-r-- 1 niusha niusha 228 May 24 22:29 Q1.c
-rw-rw-r-- 1 niusha niusha 133 May 17 22:21 Q2.sh
-rw-rw-r-- 1 niusha niusha 1 Jun 8 12:13 sample.c
-rw-rw-r-- 1 niusha niusha 0 May 17 22:06 'Untitled Document 1'
niusha@niusha-ygh:~/Desktop$
```

2. اسکریپتی بنویسید که عدد 4 رقمی از کاربر دریافت کند و حاصل ضرب ارقام آن را با میانگین آن جمع کرده و حاصل را چاپ کند (4 رقمی بودن اعداد را چک کنید).

```
niusha@niusha-ygh:~/Desktop$ bash digits.sh
enter a 4-digits number: 1234
final result is: 26.50
niusha@niusha-ygh:~/Desktop$ bash digits.sh
enter a 4-digits number: 567
the entered number is not valid!
niusha@niusha-ygh:~/Desktop$
```

```
Open  [icon]  digits.sh  Save  [icon]  [icon]  [icon]  [icon]
~/Desktop

1
2 #!/bin/bash
3
4
5 function is_four_digit_number() {
6     if [[ $1 =~ ^[0-9]{4}$ ]]; then
7         return 0
8     else
9         return 1
10    fi
11 }
12
13
14 read -p "enter a 4-digits number: " num
15
16
17 if ! is_four_digit_number "$num"; then
18     echo "the entered number is not valid!"
19     exit 1
20 fi
21
22
23 digits=$(echo $num | grep -o .)
24
25
26 product=1
27 sum=0
28 for digit in "${digits[@]}; do
29     product=$((product * digit))
30     sum=$((sum + digit))
31 done
32
33
34 average=$(echo "scale=2; $sum / 4" | bc)
35
36
37 result=$(echo "scale=2; $product + $average" | bc)
38
39
40 echo "final result is: $result"
41
```

3. الگوریتم زمان بندی مبتنی بر Round-Robin به زبان C بنویسید، به طوری که فرآیندها از لحاظ ترتیب اجرا بر اساس الویت تعیین شده اجرا میشوند.

```
niusha@niusha-ygh:~/Desktop$ ls
a.out      grp_names.txt  Q2.sh      sample.c
digits.sh  Q1.c           Round_Robin.c 'Untitled Document 1'
niusha@niusha-ygh:~/Desktop$ gcc Round_Robin.c
niusha@niusha-ygh:~/Desktop$ sudo ./a.out
Enter the number of processes: 3
Enter burst time, priority and process ID for process 1: 5 2 1
Enter burst time, priority and process ID for process 2: 3 3 2
Enter burst time, priority and process ID for process 3: 4 1 3
Enter time quantum: 1
Process 3 executed for 1 units. Remaining time: 3
Process 1 executed for 1 units. Remaining time: 4
Process 2 executed for 1 units. Remaining time: 2
Process 3 executed for 1 units. Remaining time: 2
Process 1 executed for 1 units. Remaining time: 3
Process 2 executed for 1 units. Remaining time: 1
Process 3 executed for 1 units. Remaining time: 1
Process 1 executed for 1 units. Remaining time: 2
Process 2 executed for 1 units. Completed.
Process 3 executed for 1 units. Completed.
Process 1 executed for 1 units. Remaining time: 1
Process 1 executed for 1 units. Completed.
niusha@niusha-ygh:~/Desktop$
```

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

typedef struct {
    int process_id;
    int burst_time;
    int priority;
    int remaining_time;
} Process;

void sort_by_priority(Process* processes, int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (processes[j].priority > processes[j + 1].priority) {
                Process temp = processes[j];
                processes[j] = processes[j + 1];
                processes[j + 1] = temp;
            }
        }
    }
}

void round_robin(Process* processes, int n, int quantum) {
    int time = 0;
    int done;
```

```

do {
    done = 1;
    for (int i = 0; i < n; i++) {
        if (processes[i].remaining_time > 0) {
            done = 0;
            if (processes[i].remaining_time > quantum) {
                time += quantum;
                processes[i].remaining_time -= quantum;
                printf("Process %d executed for %d units. Remaining time: %d\n",
processes[i].process_id, quantum, processes[i].remaining_time);
            } else {
                time += processes[i].remaining_time;
                printf("Process %d executed for %d units. Completed.\n",
processes[i].process_id, processes[i].remaining_time);
                processes[i].remaining_time = 0;
            }
        }
    }
} while (!done);
}

int main() {
    int n, quantum;

    printf("Enter the number of processes: ");
    scanf("%d", &n);

    Process* processes = (Process*)malloc(n * sizeof(Process));

    for (int i = 0; i < n; i++) {
        printf("Enter burst time, priority and process ID for process %d: ", i + 1);
        scanf("%d %d %d", &processes[i].burst_time, &processes[i].priority,
&processes[i].process_id);
        processes[i].remaining_time = processes[i].burst_time;
    }
    printf("Enter time quantum: ");
    scanf("%d", &quantum);
    sort_by_priority(processes, n);
    round_robin(processes, n, quantum);
    free(processes);
    return 0;
}

```

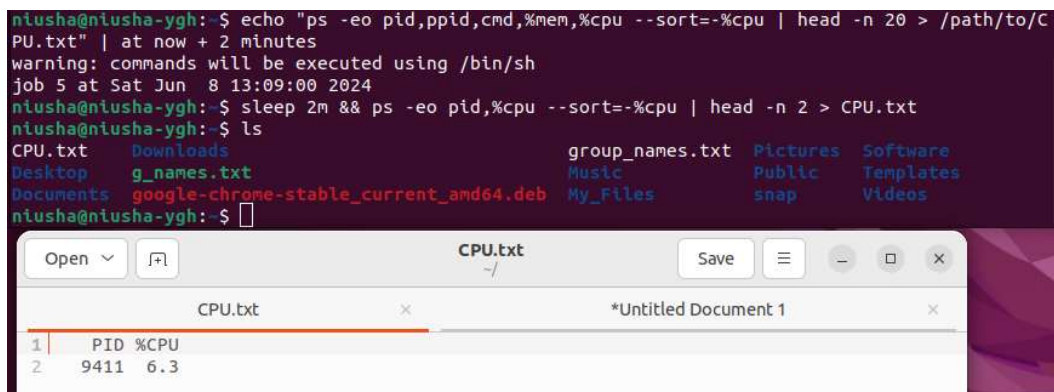
```
Open [icon] Round_Robin.c ~/Desktop Save [icon] [icon] [icon] [icon]
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 typedef struct {
5     int process_id;
6     int burst_time;
7     int priority;
8     int remaining_time;
9 } Process;
10
11 void sort_by_priority(Process* processes, int n) {
12     for (int i = 0; i < n - 1; i++) {
13         for (int j = 0; j < n - i - 1; j++) {
14             if (processes[j].priority > processes[j + 1].priority) {
15                 Process temp = processes[j];
16                 processes[j] = processes[j + 1];
17                 processes[j + 1] = temp;
18             }
19         }
20     }
21 }
22
```

```
Open [icon] Round_Robin.c ~/Desktop Save [icon] [icon] [icon] [icon]
18     }
19 }
20 }
21 }
22
23 void round_robin(Process* processes, int n, int quantum) {
24     int time = 0;
25     int done;
26
27     do {
28         done = 1;
29         for (int i = 0; i < n; i++) {
30             if (processes[i].remaining_time > 0) {
31                 done = 0;
32                 if (processes[i].remaining_time > quantum) {
33                     time += quantum;
34                     processes[i].remaining_time -= quantum;
35                     printf("Process %d executed for %d units. Remaining time: %d\n",
36                        processes[i].process_id, quantum, processes[i].remaining_time);
37                 } else {
38                     time += processes[i].remaining_time;
39                     printf("Process %d executed for %d units. Completed.\n",
40                        processes[i].process_id, processes[i].remaining_time);
41                     processes[i].remaining_time = 0;
42                 }
43             }
44         } while (!done);
45 }
```

```
Open  Round_Robin.c  Save  -  □  ×
~/Desktop
processes[i].process_id, quantum, processes[i].remaining_time);
36     } else {
37         time += processes[i].remaining_time;
38         printf("Process %d executed for %d units. Completed.\n",
processes[i].process_id, processes[i].remaining_time);
39         processes[i].remaining_time = 0;
40     }
41 }
42 }
43 } while (!done);
44 }
45
46 int main() {
47     int n, quantum;
48
49     printf("Enter the number of processes: ");
50     scanf("%d", &n);
51
52     Process* processes = (Process*)malloc(n * sizeof(Process));
53
54     for (int i = 0; i < n; i++) {
55         printf("Enter burst time, priority and process ID for process %d: ", i + 1);
56         scanf("%d %d %d", &processes[i].burst_time, &processes[i].priority,
&processes[i].process_id);
57         processes[i].remaining_time = processes[i].burst_time;
58     }
59
60     printf("Enter time quantum: ");
61     scanf("%d", &quantum);
62
63     sort_by_priority(processes, n);
64     round_robin(processes, n, quantum);
65
66     free(processes);
67     return 0;
68 }
69
C  Tab Width: 8  Ln 1, Col 1  INS
```

4. دستوری بنویسید که 2 دقیقه بعد فرآیندهایی که بیشترین میزان مصرف cpu را داشتند و در فایل به نام CPU.txt ذخیره کنید.

```
niusha@niusha-ygh: $ echo "ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%cpu | head -n 20 > /path/to/CPU.txt" | at now + 2 minutes
warning: commands will be executed using /bin/sh
job 5 at Sat Jun  8 13:09:00 2024
niusha@niusha-ygh: $ sleep 2m && ps -eo pid,%cpu --sort=-%cpu | head -n 2 > CPU.txt
niusha@niusha-ygh: $ ls
CPU.txt      Downloads      group_names.txt  Pictures      Software
Desktop      g_names.txt    Music            Public        Templates
Documents    google-chrome-stable_current_amd64.deb  My_Files      snap          Videos
niusha@niusha-ygh: $
```



The file manager window shows the following content in CPU.txt:

	PID	%CPU
1		
2	9411	6.3

5. فایل های با پسوند .txt را در دایرکتوری /etc پیدا کرده و فایل هایی که نام آن ها با حروف a-k تمام میشود را لیست کنید.

```
niusha@niusha-ygh:~/Desktop$ find /etc -type f -name "*.txt" -regex ".*[/^]*[a-k]\.txt" -print
/etc/brltty/Input/ec/spanish.txt
find: '/etc/polkit-1/localauthority': Permission denied
find: '/etc/cups/ssl': Permission denied
/etc/X11/rgb.txt
find: '/etc/ssl/private': Permission denied
niusha@niusha-ygh:~/Desktop$ find /etc -type f -name '*.txt' | grep -E '/etc/*[a-k]\.txt$'
find: '/etc/polkit-1/localauthority': Permission denied
find: '/etc/cups/ssl': Permission denied
/etc/brltty/Input/ec/spanish.txt
find: '/etc/ssl/private': Permission denied
/etc/X11/rgb.txt
niusha@niusha-ygh:~/Desktop$
```


6. در قالب یک دستور نام کاربرانی که با حرف S شروع میشوند را از فایل /etc/passwd خوانده و تعداد حروف آن را بشمارید.

```
niusha@niusha-ygh:~$ grep '^s' /etc/passwd | cut -d: -f1 | awk '{print length($0)}' |  
paste -sd+ - | bc  
96  
niusha@niusha-ygh:~$
```

7. اسکریپتی بنویسید که آدرس IP اینترفیس سیستم را دریافت و آدرس فیزیکی آن را چاپ کند.

```
niusha@niusha-ygh:~$ ls
CPU.txt  Documents  g_names.txt  group_names.txt  Music
Pictures snap  Templates
Desktop  Downloads  google-chrome-stable_current_amd64.deb  mac_add.sh  My_Files
Public   Software  Videos
niusha@niusha-ygh:~$ bash mac_add.sh
Interface: enp0s3
IP Address: 10.0.2.15/24
MAC Address: 08:00:27:8a:43:f6
-----
Interface: lo
IP Address: 127.0.0.1/8
MAC Address: 00:00:00:00:00:00
-----
niusha@niusha-ygh:~$
```

```
mac_add.sh
Save

CPU.txt  mac_add.sh
1 #!/bin/bash
2
3
4 interfaces=$(ip -o -4 addr list | awk '{print $2}' | sort | uniq)
5
6 for interface in $interfaces; do
7
8     ip_address=$(ip -o -4 addr show $interface | awk '{print $4}')
9
10
11     mac_address=$(cat /sys/class/net/$interface/address)
12
13
14     echo "Interface: $interface"
15     echo "IP Address: $ip_address"
16     echo "MAC Address: $mac_address"
17     echo "-----"
18 done
19
```