

*Course project: Operating Systems*

***Write a **shell script** that produces  
a **file of sequential numbers*****

---

Gabriela Milusheva

# Task Description

Напишете shell script, който създава файл с последователни числа, като чете последното число във файла, добавя 1 към него и след това добавя новото число към файла. Изпълнете един екземпляр на скрипта в background режим и един екземпляр на скрипта във foreground режим, всеки с достъп до един и същ файл.

- ❖ Колко време отнема, преди да се прояви race condition?
- ❖ Коя е критичната секция?
- ❖ Променете скрипта, за да предотвратите race condition. (Hint: използвайте In file file.lock, за да заключите файла с данни).

# Инсталиране на **MINIX** върху **виртуална машина**

Инсталираме ОС **MINIX** на виртуална машина,  
като използваме **Oracle VM VirtualBox**:

1. Изтегляме **ISO образа** от официалния сайт на **MINIX**.
2. Създаваме нова VM във **VirtualBox**.
3. Прикачваме **MINIX ISO образа** като виртуален диск на **VM**.
4. Стартираме **VM** и следваме инструкциите за инсталиране на **MINIX**.
5. Рестартираме **VM** и зареждаме ОС **MINIX**.



# Команди за създаване, редактиране и изпълняване на *shell script* файл в *MINIX*

Действие	Команда
Инсталиране пакета за nano editor	<code>pkgin install nano</code>
Създаване на празен shell script файл	<code>touch file_name.sh</code>
Отваряне на shell script файл в nano	<code>nano file_name.sh</code>
Изпълняване на скрипта във <i>foreground</i> режим	<code>sh file_name.sh</code>
Изпълняване на скрипта в <i>background</i> режим	<code>sh file_name.sh &amp;</code>

```
# touch file_name.sh  
# nano file_name.sh_
```

```
nano 2.6.0 File: file_name.sh  
  
#!/bin/bash  
echo "Type name:"  
read name  
echo "Hello, $name"  
  
[ Read 7 lines ]  
^G Get Help ^O Write Out ^W Where Is ^K Cut Text  
^X Exit ^R Read File ^N Replace ^U Uncut Text
```

```
# sh file_name.sh
```

```
Type name:  
World  
Hello, World!
```

## race.sh file

```
2 #!/bin/bash
3
4 echo "--> Start race.sh"
5
6 #Check if the file existsp if not, create it and write 1 to it
7 if test ! -f numbers_race
8 then
9     echo "Create the numbers_race file"
10    echo 1 > numbers_race
11 fi
12
13 echo "Repeat 100 times - read and increase last number"
14 for i in $(seq 1 100);
15 do
16     # Read and assign last line number to the variable LASTNUM
17     LASTNUM=$(tail -1 numbers_race)
18     #Increment the value of LASTNUM by 1
19     LASTNUM=$((LASTNUM + 1))
20
21     #The new value of LASTNUM is added to the end of the "numbers_race" file
22     echo $LASTNUM >> numbers_race
23 done
24 echo "--> Finish race.sh"
```

## race\_start.sh file

```
2 #!/bin/bash
3
4 #Overwriting (in this case "cleaning") the file
5 echo "Start cleaning numbers_race file..."
6 > numbers_race
7 echo "File is clean!"
8
9 echo -e "\n...Start the two race programs at the same time to see the race"
10 #Running the race.sh script in the background (&)
11 sh race.sh &
12 #Then immediately running the race.sh script in the foreground
13 sh race.sh
14
15 #Wait for 3 seconds before continuing
16 sleep 3s
17
18 echo "...Stop the two race programs at the same time to see the race"
19
20 #Exit the script with a status of 0, indicating success
21 exit 0
```

Race condition настъпва, когато две или повече нишки могат да достъпят споделени данни и се опитват да ги променят едновременно.

Thread scheduling алгоритъма може да превключва между тях по всяко време, поради което не можем да знаем реда, в който нишките ще опитат да достъпят споделените данни. Следователно резултатът от промяната на данните зависи от thread scheduling алгоритъма.

Като стартираме `race_start.sh`, можем да видим, че и двете нишки "се състезават" за достъп или промяна на данните.

Проблемът възниква, когато:

- първата нишка извършва check-then-act:
  - извършва check-1 и получава стойността на `LASTNUM`
  - след това извършва act-1 (увеличава `LASTNUM` и го добавя към файлът `numbers_race`)
- втората нишка извършва check-2 и act-2 спрямо стойността в `numbers_race` между check-1 и act-1.

# To summarize

## QUESTION 1

Колко време отнема, преди да се прояви race condition?

## ANSWER

File: numbers\_race

```
nano 2.6.0
1
1
2
2
3
3
4
4
5
5
6
6
7
7
8
8
9
9
10
10
91
91
92
92
93
93
94
94
95
95
96
96
97
97
98
98
99
99
100
100
-
```

[ Read 200 lines ]

## QUESTION 2

Коя е критичната секция (critical section)?

## ANSWER

Критичната секция/ регион е частта от програмата, където се достъпва споделената памет.

nano 2.6.0

File: race.sh

```
#!/bin/bash
_
echo "--> Start race.sh"
if test ! -f numbers_race
then
    echo "Create the numbers_race file"
    echo 1 > numbers_race
fi

echo "Repeat 100 times - read and increase last number"
for i in $(seq 1 100);
do
    #Read and increase last number
    LASTNUM=$(tail -1 numbers_race)
    LASTNUM=$((LASTNUM + 1))

    echo $LASTNUM >> numbers_race
done

echo "--> Finish race.sh"
```



## no\_race.sh file

```
2 #!/bin/bash
3
4 echo "--> Start no_race.sh"
5
6 #Check if the file exists if not, create it and write 1 to it
7 if test ! -f numbers_no_race
8 then
9     echo "Create the numbers_no_race file"
10    echo 1 > numbers_no_race
11 fi
12
13 echo "Lock numbers_no_race and do not let interruption"
14 #Attempts to lock the file to prevent other processes from modifying it
15 #If the creation of the lock file is successful, the script proceeds to execute the "for" loop
16 if ln numbers_no_race numbers_no_race.lock
17 then
18     echo "Repeat 100 times - read and increase last number"
19     for i in $(seq 1 100);
20     do
21         # Read and assign last line number to the variable LASTNUM
22         LASTNUM=$(tail -1 numbers_no_race)
23         #Increment the value of LASTNUM by 1
24         LASTNUM=$((LASTNUM + 1))
25
26         #The new value of LASTNUM is added to the end of the "numbers_no_race" file
27         echo $LASTNUM >> numbers_no_race
28     done
29
30     echo "Unlock numbers_no_race"
31     #Remove the lock file
32     rm numbers_no_race.lock
33 fi
34 echo "--> Finish no_race.sh"
```

## no\_race\_start.sh file

```
2 #!/bin/bash
3
4 #Overwriting (in this case "cleaning") the file
5 echo "Start cleaning numbers_race file..."
6 > numbers_race
7 echo "File is clean!"
8
9 echo -e "\n...Start the two no_race programs at the same time to see the race"
10 #Running the no_race.sh script in the background (&)
11 sh no_race.sh &
12 #Then immediately running the no_race.sh script in the foreground
13 sh no_race.sh
14
15 #Wait for 3 seconds before continuing
16 sleep 3s
17
18 echo "...Stop the two no_race programs at the same time to see the race"
19
20 #Exit the script with a status of 0, indicating success
21 exit 0
```

## Предотвратяване на *race condition*

File: numbers\_no\_race

nano 2.6.0

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20

90  
91  
92  
93  
94  
95  
96  
97  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100

[ Read 100 lines ]

# Исполнение на *race\_start.sh* и *no\_race\_start.sh*

## start.sh file

```
2 #!/bin/bash
3
4 #Script that starts race_start.sh and no_race_start.sh in sequence
5 echo "*****START*****"
6
7 #&& operators- ensure that the next command is only run if the previous command was successful
8
9 sh race_start.sh &&
10
11 echo -e "\n\n\n\n\n"
12
13 sh no_race_start.sh &&
14
15 echo "*****FINISH*****"
```

```
# sh start.sh
```



```
*****START*****
Start cleaning numbers_race file...
File is clean!

...Start the two race programs at same time to see the race
--> Start race.sh
Repeat 100 times - read and increase last number
--> Start race.sh
Repeat 100 times - read and increase last number
--> Finish race.sh
--> Finish race.sh
...Stop the two race programs at same time to see the race

Start cleaning numbers_no_race file...
File is clean!

...Start the two no_race programs at the same time
--> Start no_race.sh
Lock numbers_no_race and don't let interruption
--> Start no_race.sh
Lock numbers_no_race and don't let interruption
Repeat 100 times - read and increase last number
ln: numbers_no_race.lock: File exists
--> Finish no_race.sh
Unlock numbers_no_race
--> Finish no_race.sh
...Stop the two no_race programs at the same time
*****FINISH*****
```



***Thank you for the  
attention***



[github.com/Gabriela-Milusheva/OS\\_race-task](https://github.com/Gabriela-Milusheva/OS_race-task)