Студент (име и презиме):
Број индекса:
25. август 2025.

Факултет инжењерских наука
Практикум из рачунарских алата
Колоквијум

Задаци:
У свим задацима важи претпоставка да се наредбе/програми/скрипте извршавају у `bash` окружењу. Наредбе које извршавају корисници обавезно започињати са `$`, а које извршава суперкорисник са `#`.

Задатак 1 – [по 2% за сваки тачан од десет одговора, односно 20% укупно]

[2%] — Телевизијска серија *Бољи живот* снимана је у три сезоне при чему свака има по 27 епизода. Једном наредбом направити осамдесет једну празну датотеку под именима `БољиЖивотCX_EYZ.mkv` замењујући `X` тачно једном цифром (1 или 2 или 3) која означава редни број сезоне, и `YZ` цифрама (тачно две) које означавају редни број сваке епизоде (почевши од 01 па све до закључно са 27).

решење

```
$ touch БољиЖивотC{1..3}_E{01..27}.mkv
```

решење

[2%] — Направити девет празних датотека `семестралниРадПоглављеX.tex`, а `X` је једноцифрен природан број.

решење

```
$ touch семестралниРадПоглавље{1..9}.tex
```

решење

[2%] — Направити одједном три поддиректоријума `сезона1` и `сезона2` и `сезона3` у директоријуму `БољиЖивиот`.

решење

```
$ mkdir -p БољиЖивот/сезона{1..3}
```

решење

[2%] — Преместити датотеке БољиЖивотCX_EYZ.mkv у одговарајуће поддиректоријуме помоћу три наредбе.

решење

```
$ mv БољиЖивотC1_E*.mkv БољиЖивот/сезона1/
$ mv БољиЖивотC2_E*.mkv БољиЖивот/сезона2/
$ mv БољиЖивотC3_E*.mkv БољиЖивот/сезона3/
```

решење

[2%] — Једном наредбом направити директоријум `семестралниРад` у директоријуму `Документи` који садржи и један поддиректоријум `поглавља` независно од тога да ли неки од директоријума већ постоји. Затим у `Документи/семестралниРад` направити још три поддиректоријума: `ментор`, `измене` и `распуст`, такође употребом једне наредбе.

решење

```
$  mkdir -p Документи/семестралниРад/поглавља
$  mkdir Документи/семестралниРад/{ментор,измене,распуст}
```

решење

[2%] — Сада пређи у директоријум `Документи/семестралниРад/поглавља` и преместити сва поглавља у њега.

решење

```
$ cd Документи/семестралниРад/поглавља
$ mv ../../../семестралниРадПоглавље*.tex .
```

решење

[2%] — Прва три поглавља семестралног рада су послата ментору на рецензију. Како не бисте случајно направили измене у тим поглављима, преместите их у поддиректоријум `ментор` једном наредбом. Последња три поглавља семестралног рада биће написана на одмору па их пребацити у `распуст`, такође једном наредбом.

решење

```
$ mv семестралниРадПоглавље{1..3}.tex ../ментор/
$ mv семестралниРадПоглавље{7..9}.tex ../распуст/
```

решење

[2%] — Средња три поглавља семестралног рада захтевају одређене козметичке промене па их прелазећи најпре у родитељски директоријум ископирати у директоријум `измене` тако чувајући оригинале.

решење

```
$ cd ..
$ cp поглавља/семестралниРадПоглавље{4..6}.tex измене/
```

решење

[2%] — Избрисати директоријум `распуст` и целокупан његов садржај користећи рекурзивно брисање.

решење

```
$ rm -r распуст
```

решење

[2%] — Један од колега се дочепао ваше тастатуре док сте се окренули у жељи да узмете флашицу воде. Срећом, били сте хитри, па је колега стигао да унесе само једну команду, којом је променио радни директоријум. Како вас је дело колеге поприлично потресло, заборавили сте у ком сте се радном директоријуму налазили до малочас. Једном командом се пребаците у претходни радни директоријум.

решење

```
$ cd -
```

решење

Задатак 2 – [50%]

[20%] — Написати `bash` скрипту која као једини аргумент прима путању ка одређеном директоријуму, а затим у оквиру тако задатог задатог директоријума прави три поддиректоријума, и то `mali`, `srednji` и `veliki`. Након тога скрипта премешта све датотеке из прослеђеног јој директоријума које се могу налазити на произвољној дубини унутар њега пратећи следећа три једноставна правила:

- ако је величина датотеке мања од једног мегабајта у директоријум `mali`;
- ако је величина датотеке већа од једног гигабајта у директоријум `veliki`;
- иначе, ако је величина између једног мегабајта и једног гигабајта у директоријум `srednji`.

Коначно, скрипта брише све празне поддиректоријуме у задатом директоријуму, уколико их има.

решење

```
#!/usr/bin/env bash

if [[ $# -ne 1 ]]; then
    echo "Greska! Broj argumenata mora biti 1."
    exit 1
fi

if [[ ! -e $1 ]]; then
    echo "Greska! Direktorijum $1 ne postoji."
    exit 2
fi

cd "$1"

mkdir mali srednji veliki

# Представљање у бајтовима (јер наредба заокружује на мању
# целобројну вредност коришћене јединице, где ће нпр. -size -1G
# вратити датотеке које заузимају 0G тј. празне, а то не желимо):
find . -type f -size -1048576c -exec mv '{}' ./mali ';'
find . -type f ! -size -1048576c -and ! -size +1073741824c -exec  mv '{}' ./srednji ';'
find . -type f -size +1073741824c -exec mv '{}' ./veliki ';'

# Примера ради, следеће наредбе НЕЋЕ радити како треба у неким случајевима:
# find . -type f -size -1M -exec mv '{}' ./mali ';'
```

```
# find . -type f \( -size +1M -a -size -1G \) -exec mv '{}' ./srednji ';'
# find . -type f -size +1G -exec mv '{}' ./veliki ';'

# Брисање празних поддиректоријума
find . -type d -empty -exec rmdir '{}' ';'

```

решење

[20%] — Написати `bash` скрипту која као прима тачно два аргумента, и то групу фајлова над којом врши обраду (`mali`, `srednji`, `veliki`), као први и идентификатор функције коју извршава (`1, 2, 3`) као други. Функције раде по следећем принципу:

- 1 - Пребројава и у терминалу исписује број извршних фајлова унутар одговарајућег поддиректоријума на основу првог аргумента.
- 2 - Креира документ назива *grupa*_spisak.txt и у њега уписује називе свих фајлова који припадају групи коју задаје корисник.
- 3 - Брише све фајлове из одабране групе које тренутни корисник не може прочитати.

решење

```
#!/bin/bash

# Позив: bash operacije_nad_grupom.sh <grupa> <funkcija>
# <grupa>: mali | srednji | veliki
# <funkcija>: 1 | 2 | 3

if [[ $# -ne 2 ]]; then
    echo "Greska! Broj argumenata mora biti 2."
    exit 1
fi

GRUPA=$1
N=$2

if [[ $GRUPA != mali && $GRUPA != srednji && $GRUPA != veliki ]]; then
    echo "Greska! Prvi argument mora jedan od: mali, srednji, veliki."
    exit 2
fi

if [[ $N != 1 && $N != 2 && $N != 3 ]]; then
      echo "Greska! Drugi argument mora jedan od: 1, 2, 3."
      exit 3
fi

if [[ ! -d $GRUPA ]]; then
    echo "Greska! \"$GRUPA\" nije postojeci direktorijum."
    exit 4
fi

# 1. Број извршних фајлова
f1(){
    count=0
    for datoteka in $GRUPA/*; do
        if [[ -f $datoteka ]] && [[ -x $datoteka ]]; then
            ((count++))
        fi
    done
    echo "Broj izvrsnih datoteka unutar direktorijuma \"$GRUPA\" je: $count."
}

# 2. Списак свих фајлова у једном txt фајлу
f2(){
    touch ${GRUPA}_spisak.txt
```

```
    for datoteka in $GRUPA/*; do
        if [[ -f $datoteka ]]; then
            # u slucaju da se trazi samo ime datoteke, bez putanje:
            # echo $(basename $datoteka) >> ${GRUPA}_spisak.txt
            echo $datoteka >> ${GRUPA}_spisak.txt
        fi
    done
}

# 3. Брисање свих фајлова које корисник не може прочитати
f3(){
    for datoteka in $GRUPA/*; do
        if [[ -f $datoteka ]] && [[ ! -r $datoteka ]]; then
            rm -f $datoteka
        fi
    done
}

f$N
```

решење

[10%] — Рецимо да сте успешно креирали претходне 2 скрипте и да се оне зову `sort_by_size.sh` и `operacije_nad_grupom.sh`. Налазите се у директоријуму који садржи обе скрипте. Потребно је:

- Иницијализовати `git` репозиторијум.
- Направити по грану за сваку од скрипти/функционалности.
- На обе гране додати креиране скрипте и направити комитове.
- Пребацити се на главну (`master`) грану и прикупити све измене са појединачних грана.
- Написати команду којом бисте измене из локалног репозиторијума послали на главну (`master`) грану remote репозиторијума (занемарити подешавање remote-а).

решење

```
$ git init

$ git checkout -b sortiranje-po-velicini
$ git add sort_by_size.sh
$ git commit -m "Скрипта за разврставање фајлова по величини."

$ git checkout -b operacije-nad-grupom
$ git add operacije_nad_grupom.sh
$ git commit -m "Скрипта са функцијама за рад над групама фајлова"

$ git checkout master
$ git merge sortiranje-po-velicini
$ git merge operacije-nad-grupom

$ git push origin master
```

решење

Задатак 3 – [20%]

[10%] — У директоријуму `/projekat` налази се четири поддиректоријума `frontend`, `backend`, `devops` и `docs`.

На пројекту раде 4 корисника: `vlada`, `isidora`, `laza` и `fica`. Креирати наведене кориснике и подесити лозинку за сваког од њих. Поред наведених 4 корисника, на систему постоји и корисник `grego` (администратор ФИН-а).

Уредити дозволе на систему тако да буде задовољено:

- Сви корисници имају пуне привилегије над садржајем директоријума `docs`.

- Корисник `vlada`, као тим-лидер има пуне привилегије над садржајем сва 3 технолошка директоријума.
- Корисник `isidora` има право читања садржаја свих технолошких директоријума, али има право писања и извршавања само унутар директоријума `frontend`.
- Корисник `laza` има право читања садржаја свих технолошких директоријума, али има право писања и извршавања само унутар директоријума `backend`.
- Корисник `fica` има право читања садржаја свих технолошких директоријума, али има право писања и извршавања само унутар директоријума `devops`.
- Власник свих директоријума је `grego`.

Обезбедити да сви новонастали фајлови унутар поддиректоријума пројекта припадају групи надлежној за одговарајући део пројекта.

решење

```
# groupadd frontend
# groupadd backend
# groupadd devops
# groupadd docs

# useradd vlada
# usermod -a -G frontend,backend,devops,docs vlada
# passwd vlada

# useradd isidora
# usermod -a -G frontend,docs isidora
# passwd isidora

# useradd laza
# usermod -a -G backend,docs laza
# passwd laza

# useradd fica
# usermod -a -G devops,docs fica
# passwd fica

# chown grego:frontend /projekat/frontend
# chown grego:backend  /projekat/backend
# chown grego:devops   /projekat/devops
# chown grego:docs     /projekat/docs

# chmod 2777 /projekat/docs
# chmod 2774 /projekat/frontend
# chmod 2774 /projekat/backend
# chmod 2774 /projekat/devops
```

решење

Задатак 4 – – [по 2% за сваки тачан од пет одговора, односно 10% укупно]
Радите у локалном репозиторијуму који је повезан на remote (нпр. `GitHub`).

[2%] — Претпоставити да сте на локалној грани `feature`. На remote-у је ажурирана грана `master`. Потребно је довући све измене са remote-а, али без да се оне аутоматски примене у локалном репозиторијуму.

решење

```
$ git fetch origin
```

решење

[2%] — Локална грана `feature` треба да садржи све измене из `master`, али без креирања merge commit-а. Ажурирати `feature` грану тако да су примењене измене са `master` гране, али и све локалне измене које `feature` садржи у односу на локалну верзију `master` гране.

решење

```
$ git rebase origin/master
```

решење

[2%] — Приказати историју свих commit-а на тренутној грани, са приказом гранања (графички), аутора и поруке.

решење
```
$ git log --oneline --graph --decorate --all --author
```
решење

[2%] — Приказати све remote-ове и URL-ове на које је повезан репозиторијум.

решење
```
$ git remote -v
```
решење

[2%] — Излистати разлике између тренутне гране и remote гране `master`.

решење
```
$ git diff origin/master
```
решење