Kolokwium 2: Rozwiązania:

V4

(zadania w pozostałych wariantach są podobne, więc nie umieszczam wszystkich wariantów)

Zadanie 1.

(MAX. 2 PUNKTY)

PRZYGOTOWANIE: Utwórz- w domowym- podkatalog: "K2_V4" i tam dalej pracuj;

A.(1 PUNKT)

Pobierz plik: "skrypt_V4" z katalogu:

"/opt/windows/staff/pkopersk/SysOp1/Kolo3/"

na maszynie "oceanic", używając polecenia "scp", "sftp" lub panelu "mc";

B.(1 PUNKT)

Uruchom ten skrypt w bieżacej- lub w potomnej powłoce (jeśli potrzeba-

modyfikując prawa dostępu).

Uwaga: skrypt tworzy 5 różnych plików o nazwach pasujących do pewnego wzorca,

umieszczając tam: informacje o uzytkowniku (w pierwszym pliku), datę i czas oraz

(tylko w ostatnim pliku) informacje o jądrze i architekturze sprzętowej systemu.

W razie trudności- skontaktuj się z prowadzącym lub utwórz pliki samodzielnie,

na podstawie poleceń w skrypcie.

---------- **Rozwiązanie:**

```
mkdir ~/K2
cd ~/K2
```

A.

(zakładam, że login to “user”)

```
sftp user@oceanic.wsisiz.edu.pl
cd /opt/windows/staff/pkopersk/SysOp1/Kolo2/P07-08
get kolo2_V4.txt
get skrypt_V4
```

Uwagi:

- zamiast sftp mozna było użyć scp lub wykorzystać transfer z panelu mc;

- w tym drugim przypadku należało choćby podać wybraną opcję połączenia.

B.

tu uruchamiamy skrypt w shellu potomnym, wywołujac jego ścieżkę dostępu, po zmianie praw na +x dla właściciela (wywołanie potomnego interpretera jest wymuszone przez atrybut +x) :

```
chmod u+x skrypt_V4
./skrypt_V4
```

LUB, bez zmiany prawa dostępu:

```
bash ./skrypt_V4
```

Jest TEŻ możliwość (o której nie mówiłem) nakazania interpretacji BIEŻĄCEMU shellowi (także bez zmiany praw dostępu):

```
source ./skrypt_V4
```

(dalej zakładam że nazwa podana przy uruchomieniu skryptu to AB1).

Zadanie 2.

(MAX. 4 PUNKTY)

Nie używając edytora zmodyfikuj, wg. podanego niżej opisu, zawartość ostatnio utworzonego pliku (nalezy sprawdzić który to plik), ALE wynik zapisz w NOWYM pliku o nazwie takiej jak oryginał Z DODANYM NA KOŃCU członem: "_<TWOJ_LOGIN>" (np. dla oryginału o nazwie "plik" i loginu "piotr", nowa nazwa będzie "plik_piotr");

Wprowadzon3 zmiany polegają na tym, aby (kolejno):

A.(1 PUNKT)

spacje były zastąpione znakiem nowej linii ORAZ

B.(1 PUNKT)

aby nie było żadnej pustej linii ORAZ

C.(1 PUNKT)

ostatnia linia zawierała TYLKO informację o domyślnej grupie uzytkownika, zapisaną w pierwszym z utworzonych plików.

Uwagi: - MOZLIWY dodatkowy (1 PUNKT) za krótkie, eleganckie sformułowanie;

- w A. i B. : zastosować polecenie: "tr" (oraz ewentualnie inne);
- znak nowej linii to \n ;
- w B. : można to zrobić na 2 sposoby- wystarczy wybrac jeden.

---------- **Rozwiązanie:**

Polecenie

```
ls -tr -C1
```

wyświetli najnowszy plik jako ostatni. Dla pewności, opcja -C1 powoduje wypisywanie w 1 kolumnie.

A. , B.

```
cat 0_AB1-3210 | tr “ “ \\n | grep -v ^$ > 0_AB1-3210_user
```

Uwagi:

- filtr: “^$” polecenia grep wychwytuje linie, w których początek (symbol ^) I koniec (symbol $) sąsiadują ze sobą- a wiec linie puste.

- JEDNOCZESNY odczyt I zapis do/z tego samego pliku spowoduje na wstępie nadpisanie tego pliku pustą zawartością, a w konsekwencji utratę początkowej zawartości, I w efekcie otrzymuje się pusty plik !

- możliwe było użycie innych poleceń (jak `sed` czy `awk`, o których nie mówiliśmy na zajęciach)

C.

W tym wypadku nie potrzebujemy całej linii, tylko jedno z pól (drugie w wariancie z GID)

```
cut -f2  -d “ “ \[09\]_AB1 >> AB1_user
```

Zadanie 3.

(MAX. 2 PUNKTY)

A.(1 PUNKT)

Ile jest linii we wszystkich plikach utworzonych w zadaniu 1 ?

B.(1 PUNKT)

Usun, używając możliwie NAJMNIEJSZEJ liczby poleceń, wszystkie pliki utworzone w zadaniu 1. OPRÓCZ "zmienionego" (czyli OPRÓCZ pliku z "_<TWOJ_LOGIN>" na końcu nazwy);

---------- **Rozwiązanie:**

Zwłaszcza w tym zadaniu należało wykorzystać mechanizm dopasowania nazw plików do wzorca, w szczególności aby nie uwzględniać pilku skrypt_V4, który został skopiowany. Rozwiązanie polegające na wypisaniu wprost nazw wszystkich plików spełniających kryteria jest byle jakie (I tak też punktowane)

A.

```
wc -l `ls *AB1*`
```

B.

Chodzi TYLKO o pliki UTWORZONE w Zadaniu 1, a więc pomijamy plik z końcówką _user, a TAKŻE ewentualnie istniejące tu: kolo_V4 i skrypt_V4 !

```
rm `ls *AB1* | grep -v _user$`
```

Uwagi:

- w poleceniu `grep`*, znak* `"$"` *oznacza koniec linii (i sposób podania co ma być na końcu linii);*

- w poleceniu `find`*,* `"!"` *oznacza zaprzeczenie;*

- Ewentualnie można użyć konstrukcji z `find`*, choć w proponowaneym przez wiele osób rozwiązaniu,*

```
find . -type f ! -name 0_dl07-3210_user -exec rm -rf {} \;
```

BĘDZIE KIEPSKO , gdy w katalobgu znajdą się jeszcze inne pliki.

ORAZ dlatego, że usunięty zostanie katalog bieżący (opcja "`-rf`*") !*

ZWRACAM TEŻ UWAGĘ, że powyższe polecenie NIE ZADZIAŁA bez końcowego "`\;`*" lub "*`+`*"*

Zadanie 4.

(MAX. 4 PUNKTY)

Utwórz (bez edytora) plik SPIS_1.TXT zawierający 2 kolumny tekstu, gdzie:

a.(1 PUNKT)

pierwsza kolumna jest nazwą pliku w Twoim katalogu domowym ORAZ

b.(1 PUNKT)

druga kolumna numerem i-węzła tego pliku ORAZ

c.(1 PUNKT)

kolumny oddzielone są JEDNĄ (!) spacją.

d.(1 PUNKT)

wyświetl zawartośc pliku posortowaną bez rozróżniania małych/dużych liter i

z separatorem # zamiast spacji

---------- **Rozwiązanie:**

Uwagi wstępne:

- Chodziło przecież o katalog DOMOWY, a NIE o bieżący !

- W katalogu domowym każdy użytkownik ma, oprócz zwykłych, także pliki "ukryte" (nazwa zaczyna się od "."), które wyświetlają się z opcją `-a` *lub* `-A` *polecenia* `ls`

- Największą trudnością jest tu kolejność kolumn: `ls -i` *wyświetla NAJPIERW nr. i-węzła, a w treści zadania miało być odwrotnie.*

A.

to załatwia punkt A.

```
ls -C1 ~/ > SPIS_1.TXT
```

B.

Tu rozwiązanie jest bardzej złożone- obecne wersje “cut” nie pozwalają na przestawianie pól I rozwiązanie będzie jak niżej. Na początek czyszczenie dotychczasowego pliku, aby zacząć od pustego, następnie przeglądamy listę plików wycinając co trzeba I zapisując do pliku końcowego:

```
echo > SPIS_1.TXT
for I in *
do
  a=`ls -di $i | cut -f1 -d “ “`
  echo $i $a >> SPIS_1.TXT
done
```

C.

Powyższe rozwiązanie automatycznie zapewnia wymóg z punktu C., ale do kompletu dodaję tu filtr przydatny w podobnych wymaganiach:

```
tr -s “ “ | tr “ “ \#
```

Uwagi: Są możliwe INNE WARIANTY rozwiązania (dla punktów A-C):

- Przykałd 1:

(1) utworzyć listę plików w pliku `A` *oraz listę i-węzłów w pliku* `B`*, a następnie*

(2) wykorzystać polecenie “`paste`*”, łączące je w kolumny, oddzielone znakiem tabulacji, \t (I trzeba zamienić go na #) jak niżej:*

```
paste B A | tr \t \# > SPIS_1.TXT
```

a na koniec usunąć pomocnicze pliku A I B;

- Przykład 2.

Innym ciekawym rozwiązaniem które zauważyłem było użycie polecenia `awk lub sed` *do przestawienia kolejności wyciętych z linii wyrazów. Nie omawialismy tego polecenia na ćwiczeniach z racji braku czasu, więc jestem nieco zdziwiony jak popularne okazało się to rozwiązanie ;-)*

- Przykład 3

Inny ciekawy wariant to użycie odpowiednich opcji polecenia `stat`

D.

Ponieważ, zgodnie z treścią zadania, wygenerowaliśmy listę z więcej niż 1 plikiem, jest sens sortowania linii (ewentualnie zamianę spacji na # można umieścić w tej linii):

```
sort -i SPIS_1.TXT | tr “ “ \# > SPIS_2.TXT
```

Zadanie 5.

---------- **Rozwiązanie:**

Wyszukiwanie plików w katalogu, to nie tylko wypisanie polecenia `ls` (chyba że "`ls -R`") - są jeszcze katalogi I ich podkatalogi itd. Należy użyć polecenia "`find`" – ma większe możliwości niż `ls -R` i te możliwości są tu potrzebne).

A.

W wielu katalogach, w tym w /tmp, są miejsca o zastrzeżonym dostępie, co spowoduje generację błędów (wypisanych na **stderr**), I nie pozwoli wyszukać wszystkich plików.

```
find /tmp -type f | wc -l
```

Bardziej wiarygodne będzie zebranie razem tych komunikatów, zapisując **stdout** I **stderr** do wspólnego pliku

```
find /tmp -type f 2>& 1 > kronika.txt
```

Uwagi:

- Spotkałem się, sprzwdzając prace, z użyciem polecenia "`sudo`" w celu (ewentualnego) zwiększenia praw dostępu ALE NIE JEST TO poprawne (w tym zadaniu), gdyż: (1) może to być obejście zasadniczego problemu z jaką mieli się Państwo zmierzyć w zadaniu- brak praw dostępu I wynikający stąd komunikat o błędzie; (2) nie mówilismy o tym poleceniu na zajęciach, a nie jest pewne że w dowolnym systemie będziemy mieć uprawnienia; (3) I tak nie zabezpieczy nas w 100% przed wszystkimi problemami z brakiem dostępu.

(Ewentualnie, O ILE zna się hasło dla konta root, można zamiast tego wykonać polecenie `su -c "find …"` z odpowiedniki opcjami, ALE nie o to chodziło w zadaniu)

B.

```
find /dev -type b -mtime +7 -exec ls -i {} \;
```

Zadanie 6

(MAX. 5 PUNKTOW)

Ustaw domyślny znak zachęty tak, aby zawierał:

A.(2 PUNKTY)

kod zakończenia ostatnio wykonanego polecenia ORAZ czas (!bez daty);

B.(3 PUNKTY)

Poniższa sekwencja poleceń generuje pseudolosową liczbę całkowitą z zakresu 0-255:

```
od A none -t u1 -N1 /dev/urandom
```

a następna generuje 2 pseudolosowe znaki ascii (ich nazwy, gdy są one "niedrukowalne"):

```
od -A none -a N2 /dev/urandom
```

Korzystając z tej wiedzy oraz ze znanych poleceń i mechanizmów shella bash, należy:

podać ciąg poleceń AUTOMATYCZNIE generujących w bieżącym katalogu pliki o przypadkowych nazwach (według wzorca: "3 znaki ascii", "podkreślenie" i "2 cyfry" (np. +Az(_12) tak długo, dopóki wtgenerowana licaba nie jest większa niż 100.

C.(1 PUNKT)

Co zrobić, aby liczba w nazwie miała określony zakres, między (np. 10..20) ?

Uwagi: - w (A) manual shella, paragraf PROMPTING; środowisko użytkownika;

- w (b,c) polecenia: `od, test, let, if ... , while ...` lub podobne.

---------- **Rozwiązanie:**

A.

Korzystamy z predefiniowanego `\t` (oraz `\$`, dla estetyki) oraz ze zmiennej środowiskowej `“?”`, ALE musimy zacytować wywoływanie jej wartości, aby shell nie podstawił w to miejsce bieżącej wartosci na stałe, I stąd znak ‘, zamiast “

```
PS1=’$? \t\$ ‘
```

Uwaga: doświadczalnie mozna sprawdzić, że `PS1` ***nie jest*** *zmienną środowiskową I polecenie export nie zadziała tak jakby się mogło wydawać. Aby zmiana była trwał, trzeba ja wpisać do pliku konfiguracyjnego .bashrc*

B.

Można zaobserwować, że liczba zwracana przez od w pierwszej linijce, przyjmuje wartosci od 0 do 255, a więc może być 3-cyfrowa, więc nie musimy zmieniać tej linijki I używamy jej do otrzymania kawałka potrzebnego napisu, zmiennej n.

```
let n=0
while [[ $n -lt 100 ]]
do
  n=`od -A none -t u1 -N1 /dev/urandom | tr -s “ “ | tr -d “ “`
  a=`od -A none -a -N2 /dev/urandom | tr -s “ “ | tr -d “ “ | cut -c-2`
  touch ${a}_${n}
done
```

C.

Aby liczba znalazła się w zakresie od X do Y (gdzie X, Y to liczby):

- najpierw liczymy jej resztę z dzielenia przez wartość zakresu,

- a następnie dodajemy dolna granicę

(poniżej, znak nawiasu cytujemy, aby zinterpretował go dopiero “let”, a nie shell)

```
let n=n%\(Y-X\)+X
```

w szczególności. dla X=10, Y=20:

```
let n=n%10+10
```