Задача 2. (10 т.) Текстов файл с име comproc1 съдържа зададената по-долу последователност от команди на bash за Linux. Напишете вдясно какво ще бъде изведено на стандартния изход след стартиране на файла с команден ред

```
bash comproc1 ab cd ef
```

ако на стандартния вход бъде подадена следната последователност от символи 1 2

```
1       count=1
2       for i in 5 1 4 2
3       do for j
4               do if test $i -ge $#
5                       then count=` expr $count \* $i`
6                               echo $count $j >> f1
7                       else while true
8                               do echo $*
9                                       break 3
10                              done
11                      fi
12              done
13      done
14      read k1 k2
15      while cat f1 | grep $k2
16      do set $k1 $count
17              shift
18              echo $2
19              echo $1 $i
20              exit
21      done
22      echo FIN
```

Решение.

Въпреки че в началото на текстовия документ липсва означението #!/bin/bash, което оказва с какъв shell да се изпълни скрипта, това все пак е валиден скрипт и при изпълнението му с bash, ще се изпълни именно с Bash shell.

- На ред 1 се инициализира променлива count със стойност 1.
- На ред 2 започва цикъл с брояч i, който последователно ще се изпълни за i=5, i=1, i=4 и i=2.
- На ред 3 започва втори вложен цикъл, който имплицитно ще цикли по подадените аргументи на скрипта. **for j** е еквивалентно на **for j in "$@"** . $@ е масив от аргументите на скрипта.
- На ред 4 се тества стойността на променливата i, която е брояча на най-външния цикъл, дали е по-голяма или равна (името на командата **-ge** идва от greater or equal) на броя на аргументите на скрипта (**$#** е броя на аргументите, с които е стартиран скрипта). Тъй като i първоначално има стойност 5, то 5 е по-голямо от 3 (тъй като скрипта се стартира с три аргумента - ab cd и ef) и влиза в първия блок на if конструкцията на ред 5.
- На ред 5 стойността на променливата count се актуализира като се умножи по 5. **expr** е команда, която изчислява алгебричен израз за целочислено деление, деление по модул, умножение, събиране или изваждане. „Ескейпваме“ специалния символ * чрез \, за да укажем на командата expr, че това е умножение, а не регулярния символ за всичко/всяко. Тъй като израза е в наклонени единични кавички, то той ще се изпълни и стойността на променливата count ще се актуализира на 5.
- На ред 6 тази променлива count, която вече е равна на 5 ще се запише във файл с име f1 последвана от празен интервал и първия аргумент след него. Текущо съдържание на f1:

5 ab

- Връщаме се отново на ред 3 като този път взимаме втория аргумент на скрипта, който е cd. На ред 4, i отново е 5 и отново е по-голямо от броя на аргументите 3. На ред 5 отново актуализираме стойността на променливата count и тя вече е 25. Записваме във файла f1 новата стойност на променливата count и втория аргумент на скрипта, като ги

добавяме след наличното съдържание:

5 ab
25 cd

- Аналогично ще изциклим и последният трети аргумент на скрипта, след което стойността на променливата count ще е 125 и ще ги запишем заедно във файла f1, като той ще остане със следното съдържание:

5 ab
25 cd
125 ef

- След като сме изциклили всички аргументи на скрипта с $i равно на 5, преминаваме на $i равно на 1, което не е по-голямо от броя на аргументите, с които сме стартирали скрипта и следователно на ред 4 тестът в if конструкцията ще върне стойност false и ще влезем в else блок-а. Там имаме трети вложен цикъл (while), който е зададен като безкраен и влизайки в него ще се принтират на конзолата аргументите на скрипта (**$*** са аргументите на скрипта като един цял стринг). След това имаме **break** с аргумент 3, което означава, че ще се счупят циклите до 3-тия вложен нагоре, т.е. тъй като while е третият вложен, то ще се счупят всички цикли до този момент.
- На ред 14 прочитаме стойностите на променливите k1 и k2 от стандартния вход.
- На ред 15 се принтира на конзолата съдържанието на текстовия файл f1 и с командата **grep** с аргумент стойността на променливата k2 се проверява ред по ред дали в него някой ред съвпада с нея. Тъй като k2 има стойност равна на 2 (прочитаме 1 и 2 от стандартния вход), то ще съвпаднат втория и третия ред от файла и командата grep ще върне 0 (тоест с 0 ще индикира че е намерила поне едно съвпадение), което ще се интерпретира (контраинтуитивно) като **true** и ще продължи във while цикъла. (В случай, че grep не беше намерил нито едно съвпадение, щеше да върне 1, което щеше да се интерпретира като false).
- На ред 16 командата set ще промени параметрите подадени на скрипта и ще направи така, че първият параметър да е равен на $k1, т.е. 1, а вторият параметър да е равен на $count, т.е. 125. Третият параметър ще бъде изтрит.
- На ред 17, командата **shift** трябва да премести аргументите на скрипта наляво с броя позиции, които е приела като аргумент. В случая такъв аргумент няма и по подразбиране се взима 1. Първият аргумент вече е равен на 125, а втори няма.
- На ред 18 ще се принтира на конзолата празен ред.
- На ред 19 ще се принтира на конзолата стойността на първия аргумент, която е 125 и до нея разделена с разстояние стойността на i, която е 1.
- На ред 20 командата exit ще приключи изпълнението на скрипта и никога няма да се стигне до изпълнението на по-долните редове.

Окончателно на стандартния изход след стартиране на скрипта с команден ред

```
bach comproc1 ab cd ef
```

и подадени на стандартния вход символите 1 2
ще се изведе

ab cd ef
25 cd
125 ef

125 1

□