Programske paradigme – grupa A

1. **[Haskell 34%]** Ulazni argument je lista od N unutrašnjih podlisti i broj K. Napisati funkciju koja zadatu listu podlisti transformiše u listu od N brojeva takvih da je i-ti broj liste dobijen množenjem prvih K elemenata svake od unutrašnjih podlisti. U slučaju da je podlista kraća od K, vratiti proizvod cele podliste, a u slučaju da je lista prazna vratiti neutral za množenje, odnosno 1. Test primer:

   h1 [[**3,5**,3,1,5],[**3,-1**,6,2,-34],[**-335,3**,2],[],[**3**]] 2 = [15,-3,-1005,1,3]

2. **[Prolog 33%]** Predikat prihvata listu brojeva i sortira je opadajuće prema broju cifara u okviru svakog od brojeva. Ako dva broja imaju isti broj cifara prednost u redosledu ima onaj koji je veći. Pretpostaviti da nema negativnih brojeva u test primerima. Test primer:

   p1(4,6,66,33,5,666,3265,454,477,325,4,674,4464,4646474,3453],X):-
   X=[4646474,4464,3453,3265,674,666,477,454,325,66,33,6,5,4,4]

3. **[Java Spark 33%]** Napisati distribuirani program koji pronalazi sumu svih prostih brojeva iz datoteke brojevi1.txt (datoteka se nalazi u dokumentaciji, na putanji vezbe/11).

Napomena: Haskell funkciju i Prolog predikat imenovati sa h1 odnosno p1 kao što je gore navedeno i sačuvati ih u datotekema pod nazivom h1.hs odnosno p1.pl. Za Java Spark zadatak se ostavlja samo paket u kojem se nalazi kod. Ove dve datoteke i paket direktorijum za Spark ubaciti u direktorijum koji se zove GrupaAmiXXXYYImePrezime gde je miXXXYY zamenjeno sa oznakom naloga na alas-u. **Vreme za rad:** (**2 sata 45 minuta – vreme potrošeno na teorijskom delu).**

Programske paradigme – grupa A

1. **[Haskell 34%]** Ulazni argument je lista od N unutrašnjih podlisti i broj K. Napisati funkciju koja zadatu listu podlisti transformiše u listu od N brojeva takvih da je i-ti broj liste dobijen množenjem prvih K elemenata svake od unutrašnjih podlisti. U slučaju da je podlista kraća od K, vratiti proizvod cele podliste, a u slučaju da je lista prazna vratiti neutral za množenje, odnosno 1. Test primer:

   h1 [[**3,5**,3,1,5],[**3,-1**,6,2,-34],[**-335,3**,2],[],[**3**]] 2 = [15,-3,-1005,1,3]

2. **[Prolog 33%]** Predikat prihvata listu brojeva i sortira je opadajuće prema broju cifara u okviru svakog od brojeva. Ako dva broja imaju isti broj cifara prednost u redosledu ima onaj koji je veći. Pretpostaviti da nema negativnih brojeva u test primerima. Test primer:

   p1(4,6,66,33,5,666,3265,454,477,325,4,674,4464,4646474,3453],X):-
   X=[4646474,4464,3453,3265,674,666,477,454,325,66,33,6,5,4,4]

3. **[Java Spark 33%]** Napisati distribuirani program koji pronalazi sumu svih prostih brojeva iz datoteke brojevi1.txt (datoteka se nalazi u dokumentaciji, na putanji vezbe/11).

Napomena: Haskell funkciju i Prolog predikat imenovati sa h1 odnosno p1 kao što je gore navedeno i sačuvati ih u datotekema pod nazivom h1.hs odnosno p1.pl. Za Java Spark zadatak se ostavlja samo paket u kojem se nalazi kod. Ove dve datoteke i paket direktorijum za Spark ubaciti u direktorijum koji se zove GrupaAmiXXXYYImePrezime gde je miXXXYY zamenjeno sa oznakom naloga na alas-u. **Vreme za rad:** (**2 sata 45 minuta – vreme potrošeno na teorijskom delu).**