Zadatci za 3. laboratorijske vježbe 2018./2019.

1. Napišite funkciju čiji je prototip:
   int upis (cvor **glava, cvor **rep, float broj);

   koja kao argumente prima adrese pokazivača na početak i kraj liste (***glava** i ***rep**) i realni broj tipa float (**broj**), koji uvijek treba dodati na kraj liste. Funkcija vraća 1, ako je novi čvor uspješno dodan u listu, a 0 inače.

   Potrebno je definirati strukturu **cvor** koja sadrži broj tipa float te pokazivač na sljedeći čvor u listi. U glavnom programu potrebno je s tipkovnice učitati **n** brojeva tipa float (**n** ≤ 10) te brojeve dodati u listu korištenjem funkcije upis. Na kraju je potrebno ispisati sve članove liste počevši od početka liste te listu obrisati.

2. Strukturu podataka red potrebno je realizirati kao polje. U tu svrhu prvo je potrebno definirati strukturu **Red** koja sadrži polje realnih brojeva (od najviše 10 članova) te cijele brojeve ulaz i izlaz, koji predstavljaju indekse ulaza i izlaza u red.
   Zatim je potrebno napisati funkcije za inicijalizaciju reda, dodavanje novog podatka (**broj**) u red te za brisanje podatka (**broj**) iz reda realiziranog poljem:
   void init_red (Red *red);
   int dodajURed (double broj, Red *red);
   int skiniIzReda (double *broj, Red *red);

   U glavnom programu s tipkovnice učitati **n** realnih brojeva (**n** ≤ 10) te ih dodati u red realiziran poljem. Zatim iz reda obrisati sve članove i pri tome ih ispisati. Koristiti funkcije **dodajURed** i **skiniIzReda**.

3. Strukturu podataka red potrebno je realizirati kao listu, čiji su elementi čvorovi tipa **cvor**. Strukturu **cvor** definirati tako da sadrži realni broj i pokazivač na sljedeći čvor. Također je potrebno definirati strukturu **Red** koja sadrži pokazivače ulaz i izlaz, koji pokazuju na ulaz i izlaz iz reda realiziran listom, a čiji su elementi tipa **cvor**.
   Zatim je potrebno napisati funkcije za inicijalizaciju reda te dodavanje novog podatka (**broj**) u red realiziranog listom:

   void init_red (Red *red);
   int dodajURed (double broj, Red *red);

   U glavnom programu s tipkovnice učitati **n** realnih brojeva (**n** ≤ 10) te ih dodati u red realiziran listom pozivom funkcije **dodajURed**. Nakon što je čvor dodan u red, treba ispisati vrijednost koja je upravo dodana u red.

4. Strukturu podataka red potrebno je realizirati kao listu, čiji su elementi čvorovi tipa **cvor**. Strukturu **cvor** definirati tako da sadrži cijeli broj i pokazivač na sljedeći čvor. Također je potrebno definirati strukturu **Red** koja sadrži pokazivače ulaz i izlaz, koji pokazuju na ulaz i izlaz iz reda realiziranog listom, a čiji su elementi tipa **cvor**.
   Zatim je potrebno napisati funkciju za inicijalizaciju reda te funkciju za dodavanje novog podatka (**broj**) u red realiziranog listom:

   void init_red (Red *red);
   int dodajURed (double broj, Red *red);

   Također je potrebno napisati funkciju koja rekurzivno dodaje elemente polja **polje** od **n** članova u red (pozivati funkciju **dodajURed**) tako da se u red prvo doda zadnji element polja. Prototip funkcije je:

   int poljeURed (int polje[], int n, Red *red);

   Funkcija treba vratiti 1, ako je uspjelo dodavanje svih elemenata iz polja u red, a 0, ako nije uspjelo dodavanje nekog elementa u red. U slučaju neuspješnog dodavanja nekog elementa u red, prekinuti s dodavanjem sljedećih elemenata. Kod dodavanja svakog elementa u red ispisati element koji se upravo dodaje.

   U glavnom programu slučajnim odabirom generirati polje **10** cijelih brojeva iz intervala [1, 10] te ih dodati u red realiziran listom pozivom funkcije **poljeURed**. Ispisati članove polja prije dodavanja u red.

5. Napisati funkciju za dodavanje u stog u koji se pohranjuju cijeli brojevi. Stog treba biti implementiran poljem i maksimalna veličina mu treba biti 100. U glavnom programu stvoriti jedan stog, nasumično generirati 101 broj te njima napuniti stog.

6. Napisati funkciju za dodavanje u stog u koji se pohranjuju cijeli brojevi. Stog treba biti implementiran pokazivačima. U glavnom programu stvoriti jedan stog, nasumično generirati 101 broj te njima napuniti stog.

7. Napisati funkciju za dodavanje u stog i skidanje elemenata sa stoga. Elementi koji se pohranjuju na stog su cijeli brojevi. Stog treba biti implementiran kao polje od 10 članova.
   U glavnom programu stvoriti stog, nasumično generirati 10 cijelih brojeva iz intervala [1, 10] te njima napuniti stog, a zatim ispisati sadržaj stoga tako da se prvo ispiše element na dnu stoga, a posljednji se ispisuje element na vrhu stoga. Dozvoljeno je korištenje pomoćnog stoga.

8. Napisati funkciju za upis cijelih brojeva u hash veličine M=128. Koliziju treba riješiti korištenjem metode ulančavanja. Funkcija raspršenog adresiranja (adresa) treba biti realizirana metodom množenja pri čemu konstanta A odgovara izračunatoj vrijednosti za 32-bitne brojeve (A=2654435769). U glavnom programu nasumično generirati 129 cijelih brojeva te ih dodati u hash-tablicu pozivom funkcije za upis.

9. Napisati funkciju za upis cijelih brojeva u hash veličine M=256. Koliziju treba riješiti korištenjem kvadratnog ispitivanja. Funkcija raspršenog adresiranja (adresa) treba biti realizirana metodom množenja pri čemu konstanta A odgovara izračunatoj vrijednosti za 32-bitne brojeve (A=2654435769). U glavnom programu nasumično generirati 257 cijelih brojeva te ih dodati u hash-tablicu pozivom funkcije za upis.