Napomene:

1. Na odbranu zadatka obavezno poneti indeks ili ličnu kartu.
2. Program mora biti ručno napisan kompletno u asembleru i mora se nalaziti u jednom fajlu. Program ne sme biti trivijalan (da radi samo ispis za neki set ulaza), nego mora realizovati algoritme tražene u zadatku.
3. Naziv fajla treba da bude **vaše korisničko ime** u laboratorijama A2-1, A2-3, odnosno JUG-111. Na primer, ako je vaše korisničko ime ra123-2010, tada naziv fajla treba da bude
   ra123-2010.S
4. Za testiranje programa je dat skript **testiraj.sh**. Ovaj skript, proširen sa još nekoliko testova, će se koristiti i za testiranje na odbrani radova i od njegovog izlaza **direktno** zavisi broj osvojenih poena.
5. Poruke koje program ispisuje treba da budu **iste** kao u zadatim test primerima (vide se prilikom pokretanja skripta). Izgled ispisa (razmaci, *newline* znaci) takođe treba da bude **isti** kao u zadatim test primerima. Ispis karaktera koji nemaju vidljivu reprezentaciju (kodovi 0x1-0x1f i 0x80-0xff) će rezultovati pogrešnim ispisom, i takvi znaci će se u ispisu pojaviti u obliku [0xN], gde je N heksadecimalni kod znaka.
6. Da bi neki test uspešno prošao, očekuje se da izlaz vašeg programa bude tačno onakav kakav se traži u zadatku. Ukoliko izlaz nije isti, makar se razlikovao i u sitnijem detalju, **neće se smatrati ispravnim**.
7. Ako se program zaglavi na nekom od primera ili bude nasilno prekinut za neki od testova (segmentation fault, arithmetic exception i slično), smatraće se da program **ne prolazi nijedan test**.
8. Urađeni zadatak se donosi na USB (Flash) disku u terminu objavljenom u obaveštenju o domaćem zadatku (i to samo u tom terminu).
9. Prilikom pregledanja, vaš program će biti testiran sa još nekoliko test ulaza, koji nisu unapred zadati.
10. Prilikom pregledanja, vaš program će se uporediti sa dotad predatim rešenjima. Ukoliko se uoči značajna sličnost sa nekim prethodno predatim rešenjem, rešenje se **ne priznaje**.
11. Nakon što se zadatak pregleda, biće postavljeno par pitanja vezanih za predati kod. Ukoliko odgovori na ta pitanja ne budu zadovoljavajući, rešenje se **ne priznaje**.

Zadatak:
Napisati asemblerski program koji od korisnika traži da unese string maksimalne dužine 100 znakova. Program potom analizira string i u njemu pronalazi sve neoznačene 32-bitne heksadecimalne brojeve. Heksadecimalnim brojem se smatra niz znakova koji čine heksadecimalne cifre, pri čemu se mogu koristiti i mala i velika slova. Svi heksadecimalni brojevi moraju biti razdvojeni separatorima od ostalih reči u stringu, pri čemu se separatorom smatra svaki znak koji nije slovo engleske abecede ili dekadna cifra.

Program treba da izračuna 64-bitni zbir svih 32-bitnih heksadecmalnih brojeva (oni koji mogu da stanu u 32 bita) i da taj zbir ispiše na ekranu takođe u heksadecimalnom brojnom sistemu (koristeći velika slova). Ukoliko u stringu nema heksadecimalnih brojeva koji staju u 32 bita, ispisati odgovarajuću poruku.

Ukoliko prilikom unošenja stringa korisnik unese prazan string, treba ispisati poruku o tome i tražiti ponovno unošenje stringa (praznim stringom se smatra i onaj koji se sastoji isključivo od razmaka i tabulacija). Izlazni kod programa treba da bude 0 ako je u stringu bilo 32-bitnih heksadecimalnih brojeva, a 1 ako ih nije bilo.

Primeri upotebe:

```
Unesite string: !%$!^%!#baba1234!!! nije hex deda5678?
Zbir: 1999468AC

Unesite string:
Uneli ste prazan string!
Unesite string: 12345678abc ovde nema 32bitnih brojeva!
Nema 32-bitnih hex brojeva.
```

Za ostale primere pogledajte testiraj.sh.