Zadaci sa rešenjima dostupnim na sajtu

1. Napisati asemblerski program za poređenje dva označena broja (*a* i *b*) u **dvostrukoj preciznosti**. Rezultat rada programa treba da se nađe u registru *eax,* i to:
    - 0 ako je *a = b*
    - -1 ako je *a < b*
    - 1 ako je *a > b*
2. U asemblerskom jeziku implementirati množenje dva broja pomoću sabiranja, **koristeći promenljive**
    a. Zadatak je identičan zadatku 2 sa prvog termina vežbi; jedina razlika je što dva broja koja se množe, kao i rezultat, treba smestiti u promenljive (označene memorijske lokacije) umesto u registre; brojevi se mogu smestiti u promenljivu odmah pri inicijalizaciji (u .section .data delu koda), ne mora se koristiti *mov* instrukcija
    b. Implementirati optimizaciju kao u zadatku 2 sa prvog termina vežbi (minimalan broj iteracije petlje)
3. U asemblerskom jeziku implementirati množenje dva broja pomoću sabiranja, **koristeći promenljive dvostruke preciznosti**
    a. Osnovna ideja je identična kao i kod prethodnog zadatka
    b. Razlika: treba koristiti 64-bitne promenljive
    c. Iskoristiti primer sabiranja u dvostrukoj preciznosti dat na slajdovima sa drugog termina vežbi; proveravati grešku u prekoračenju
4. Napisati asemblerski program koji pronalazi n-ti element Fibonačijevog niza, **koristeći promenljive**
    a. Zadatak je identičan zadatku 3 sa prvog termina vežbi; jedina razlika je što broj n, kao i rezultat, treba smestiti u promenljive (označene memorijske lokacije) umesto u registre; brojevi se mogu smestiti u promenljive odmah pri inicijalizaciji (u .section .data delu koda), ne mora se koristiti *mov* instrukcija
    b. Dostupna su dva rešenja ovog zadatka; drugo rešenje (optimizovano) je kraće i koristi manje registara
5. Napisati asemblerski program koji pronalazi n-ti element Fibonačijevog niza, **koristeći promenljive dvostruke preciznosti**
    a. Osnovna ideja je identična kao i kod prethodnog zadatka
    b. Razlika: smatrati da su članovi niza (uzastopni Fibonačijevi sabirci), pa ujedno i konačan rezultat, 64-bitni; broj n može biti u jednostrukoj preciznosti
6. Napisati asemblerski program koji računa faktorijel broja n, **koristeći promenljive**
    a. Zadatak je identičan zadatku 4 sa prvog termina vežbi; jedina razlika je što broj n, kao i rezultat, treba smestiti u promenljive (označene memorijske lokacije) umesto u registre; brojevi se mogu smestiti u promenljive odmah pri inicijalizaciji (u .section .data delu koda), ne mora se koristiti *mov* instrukcija
7. U asemblerskom jeziku implementirati deljenje dva broja pomoću oduzimanja, uz čuvanje ostatka, **koristeći promenljive dvostruke preciznosti**
    a. Zadatak je identičan zadatku 5 sa prvog termina vežbi; jedina razlika je što deljenik, delilac, količnik i ostatak treba smestiti u promenljive (označene memorijske lokacije) umesto u registre; brojevi se mogu smestiti u promenljive odmah pri inicijalizaciji (u .section .data delu koda), ne mora se koristiti *mov* instrukcija

Zadaci bez rešenja dostupnih na sajtu

1. Napisati asemblerski program koji računa NZD dva broja, **koristeći promenljive**
    a. Izmeniti primer računanja NZD-a sa prvog termina vežbi, tako da se za brojeve i rezultat umesto registara koriste promenljive (inicijalizovane u .section .data delu koda, ne mora se koristiti *mov* instrukcija za početne vrednosti)
2. U asemblerskom jeziku, rezervisati jednu 32-bitnu reč iu memoriji dati joj proizvoljnu vrednost. Nakon toga, njena četiri bajta redom smestiti u registre **ah, al, bh, bl,** tako da najniži bajt bude u **ah,** a najviši u **bl**
3. U asemblerskom programu implementirati oduzimanje **dve promenljive dvostruke preciznosti**
    a. Uraditi zadatak i za označene i za neoznačene brojeve
    b. Koristiti kao pomoć primer sabiranja u dvostrukoj preciznosti sa slajdova sa drugog termina vežbi
    c. Za oduzimanje sa pozajmicom koristiti *sbb* instrukciju
    d. Obratiti pažnju na to pri oduzimanju kojih delova (viših ili nižih) može doći do greške u prekoračenju
4. Proširiti zadatke 3 i 7 iz sekcije zadataka sa rešenjima tako da rade i sa označenim brojevima
5. U asemblerskom jeziku, realizovati formulu *a = 2*b + c/3* gde su *a, b* i *c* brojevi u dvostrukoj preciznosti
    a. Smestiti *b* i *c* u promenljive
    b. Rezultat, odnosno *a*, takođe se treba naći u memoriji, odnosno promenljivoj
6. Proširiti zadatak 6 iz sekcije zadataka sa rešenjima tako da radi sa promenljivama u **dvostrukoj preciznosti**
7. Napisati asemblerski program koji računa n-ti stepen broja b
    a. Broj **b**, kao i **rezultat**, treba da budu **promenljive u dvostrukoj preciznosti**
    b. Broj **n** je **promenljiva u jednostrukoj preciznosti**