1. **Uputstvo**: Spoj svaki iskaz sa odgovarajućim pojmom! Svaki pojam sa leve strane ima odgovarajući iskaz sa desne.

1. ***Point-to-point communication***	*a) Send* rutina koja se ne vraća dok se ne kompletira(primljena ili baferovana)
2. ***Collective communication***	b) Komunikacija koja uključuje jednu ili više grupa procesa
3. ***Communication mode***	c) *Send* rutina koja se ne kompletira dok ne stigne potvrda da je primalac primio tu poruku
4. ***Blocking send***	d) Operacija u kojoj jedan proces šalje isti podatak ostalima
5. ***Synchronous send***	e) Komunikacija koja uključuje samo jedan par procesora
6. ***Broadcast***	f) Operacija u kojoj jedan proces distibuira različite elemente lokalnog niza drugima
7. ***Scatter***	g) Operacija u kojoj jedan proces prikuplja podatke sa ostalih procesa i smešta ih u lokalni niz
8. ***Gather***	h) Specifikacija metode operacije i kriterijuma kompletiranja komunikacione rutine

2. **Šta** je od navedenog tačno za sve *send* rutine?
    a) Uvek je bezbedno promeniti vrednost poslate promenljive posle *send* rutine
    b) Kompletiranje označava da je poruka primljena na odredište
    c) Uvek je bezbedno promeniti vrednost poslate promenljive nakon kompletiranja *send*-a
    d) Sve prethodno je tačno
    e) Ni jedan od prethodnih iskaza nije tačan

3. **Razmotrite** sledeći MPI pseudo kod koji neki podatak od procesa 0 do procesa 1:

```
MPI_INIT()
MPI_COMM_RANK(MPI_COMM_WORLD, myrank, ierr)
if(myrank==0)
MPI_SEND(some data to processor 1 in MPI_COMM_WORLD)
else
{
     MPI_RECV(data from processor 0 in MPI_COMM_WORLD)
     print "Message received!"
}
MPI_FINALIZE()
```

gde su MPI_SEND i MPI_RECV *blocking send* i *receive* rutine. Tako, nailazeći na MPI_RECV proces 1 se blokira dok čeka poruku od procesa 0.

Ako se kod startuje na jednom procesoru, šta očekujete da će se desiti? Objasniti.

a) Kod će otštampati "Message received!", i onda završiti.
b) Kod će se izvršiti bez izlaza.
c) Kod će se blokirati bez izlaza.
d) Doći će do greške pri izvršavanju.

4. **Šta** će se dogoditi ako se startuje na tri procesora? Objasniti.

a) Kod će otštampati "Message received!", i onda **završiti**.
b) Izvršavanje se prekida bez izlaza.
c) Kod će otštampati "Message received!", i onda **blokirati.**
d) Otštampaće *error mesage* i završiti..

5. **Pretpostavimo** da je jedini komunikator korišćen za ovaj problem MPI_COMM_WORLD. Posle poziva MPI_INIT, proces 1 odmah šalje dve poruke procesu 0. Prva poruka ima *sent tag* 100, a druga 200. Posle poziva MPI_INIT i provere da ima bar dva procesa u komunikatoru, proces 0 prima poruku sa *source* argumentom 1 i *tag* argumentom postavljenim na 200. Izaberi tačan odgovor:

a) Proces 0 je upao u deadlock, pošto pokušava da primi drugu poruku pre nego što je primio prvu.
b) Proces 0 prima drugu poruku poslatu od procesa 1 iako nije primio prvu.
c) ništa od gore navedenog

6. **Hoćemo** da uradimo jednostavno *broadcast* promenljive abc[7] na procesu 0 ka istoj lokaciji na ostalim procesima u komunikatoru. Koja je tačna sintaksa koja izvršava ovaj *broadcast*?

a) MPI_Bcast (&abc[7], 1, MPI_REAL, 0, comm)
b) MPI_Bcast (&abc, 7, MPI_REAL, 0, comm)
c) MPI_Broadcast (&abc[7], 1, MPI_REAL, 0, comm)

7. **Neka** komunikator ima 4 procesa. Koliko se ukupno MPI_SEND-a i MPI_RECV-a zahteva da bi se izvršila sledeća naredba:
MPI_ALLREDUCE (&a, &x, 1, MPI_REAL, MPI_SUM, comm) ? Objasniti, nacrtati sliku.

a) 3
b) 4
c) 12
d) 16