1. Forklar hvordan dette programmet virker, og hva det skriver ut:

Barneprosessen write-funksjonen åpner FIFO-filen, skriver meldingen og avsluttes med «\0» nullterminator. Printer meldingen som ble mottatt og lukker FIFO-filen.

Foreldreprosessen read-funksjonen åpner FIFO-filen, leser meldingen, printer det som er mottatt og lukker FIFO-filen.

I hovedprogrammet lages en FIFO-fil som gir lese og skrive tilgang til alle ved bruk av «0666».

Det lages en barneprosess med fork, returverdien er 0. I foreldreprosessen returneres barnets pid.

If barneprosesse's pid er 0 kjører write funksjonen, else leser meldingen og lukker FIFO-filen. Foreldreprosessen venter på barneprosessen med wait(NULL) og unlink sletter FIFO-filen for at det blir synkronisering.

Output:

```
Message sent    : Music from Big Pink!
Message received: Music from Big Pink!
```

3.a) Hvilket resultat får du? Forklar resultatet:

```
aleksawo@itstud:~/htdocs/oblig7$ gcc -o write_shm write_shm.c -lrt
aleksawo@itstud:~/htdocs/oblig7$ gcc -o read_shm read_shm.c -lrt
aleksawo@itstud:~/htdocs/oblig7$ ./read_shm ; ./write_shm
sum1 = 0, sum2 = 50000005000000
aleksawo@itstud:~/htdocs/oblig7$ ./write_shm ; ./read_shm
sum1 = 50000005000000, sum2 = 50000005000000
```

Det prøves å lese delte data fra minnet, men write har ikke kjørt og alle verdier er 0. For sum2 begrense svaret med formelen.

3.b)

Det skrives verdier fra 1 til 10 millioner i minnet, delte minne har riktige verdier. Read som kjøres etter skriver verdiene. Begge «sum» blir like store pga. verdiene i minne er riktige.

3.c) Kjør denne kommandoen flere ganger etter hverandre. Hvilke resultater får du? Forklar hva som skjer.

```
aleksawo@itstud:~/htdocs/oblig7$ (./write_shm &) ; ./read_shm
sum1 = 17313169274998, sum2 = 50000005000000
aleksawo@itstud:~/htdocs/oblig7$ (./write_shm &) ; ./read_shm
sum1 = 22307766180553, sum2 = 50000005000000
aleksawo@itstud:~/htdocs/oblig7$ (./write_shm &) ; ./read_shm
sum1 = 21056931351492, sum2 = 50000005000000
```

Dette skjer fordi read begynner å lese mens write fortsatt skriver til minnet. Ved å kjøre parallelt kan vi ikke vite hvilken rekkefølge operasjonene utføres.