```
/* Räkneövning 2, Uppgift 1 */

// inkludera nödvändiga bibliotek

#define MAX_LANGD 255

typedef struct adress {
        char forNamn[MAX_LANGD];
        char efterNamn[MAX_LANGD];
        char gatuAdress[MAX_LANGD];
        int postNummer;
        char postAnstalt[MAX_LANGD];
}kontakt;

int main(int argc, char *argv[]) {

        FILE *pFile = fopen("output.dat", "rb");
        /* Försök öppna fil för binär läsning */
        kontakt Adress;

        if(pFile == NULL) {
                if(errno == <errno.h>) {
                perror("Error");
                pFile = fopen("output.dat", "wb");
                /* Öppna fil för binär skrivning */

                /* Läs in adress från tangentbordet */

                fwrite(/* rätt parametrar för fwrite */);

                /* Stäng filbuffert */
                }

        }
        else {
                fread(/* rätt parametrar för fread */);

                /* Skriv ut adressen på skärmen */
        }

        return 0;
}
```

```
/* Räkneövning 2, Uppgift 2 */

// inkludera nödvändiga bibliotek

#define MAX_LANGD
#define BUFFER_SIZE

typedef struct address {...}kontakt;

int lasInAddress(kontakt *);
int stripEnd(char *);
int lasDataFilen(unsigned char * dataBuffer, FILE * file);
int skrivUtAdresser(unsigned char *dataBuffer, int antalFalt);

/* global dataBuffer */
unsigned char *dataBuffer;

int main(int argc, char *argv[]) {
        /* Pekare till filen som används för att lagra data */
        FILE *utFil;
        /* Sparar errno till denna variabel och använder den så att
        errno inte ändras mitt i koden */
        int errorNr;
        /* Variabeln för addressen */
        kontakt addressen;
        int i, antalStrukturer;

        /* Kolla att ett filnamn är givet, om inte
        avsluta programmet och printa ut instruktioner */

        /* Öppna filen som gavs in åt programmet från kommandprompten */
        if ( (utFil=fopen(argv[1], "rb")) == NULL ) {
        /* fopen returnerar NULL, kollar errno för felet*/
        /* spara errno så att inget fel uppstår om errno ändras */

                /* Om filen inte existerar →
                läs in data och skriver det till filen */

                /* Öppna filen så att man kan skriva till den */
                        if ( (utFil=fopen(argv[1], "wb")) == NULL ) {
                                /* Kunde inte öppna filen, avsluta */
                                perror("Kunde inte öppna filen för
skrivning");
                                exit(EXIT_FAILURE);
                        }
                        /* Läs in 3 olika addresser */
                        for ( i = 0; i < 3; i++ ) {
                                lasInAddress(&addressen);
                                /* Skriv addressen till filen */
```

```
                                fwrite(/* fwrite-parametrar */);
                        }
                        /* Stäng filen */
                        fclose(utFil);
                } else {
                        /* Filen existerar men kunde inte öppnas */
                        perror("Kunde inte öpna filen");
                        exit(EXIT_FAILURE);
                }
        } else {
                /* läs in data från binära filen
                och returnera antalet strukturer */
                antalStrukturer = lasDataFilen(dataBuffer, utFil);

                /* Skriv ut namnet på alla personer som har en adress */
                skrivUtAdresser(dataBuffer, antalStrukturer);

                if(antalStrukturer > 0) {
                        /* Frigör allokerat minne om det behövs */
                        free(dataBuffer);
                }
                /* Stäng filen */
                fclose(utFil);
        }
        return EXIT_SUCCESS;
}

/* Denna funktion läser in data från stdin och sparar data i addressen */
int lasInAddress(kontakt *addressen) {

  /* Fungererar med principen:
     -> printa ut instruktioner
     -> läs in data
     -> ta bort sista '\n'
     -> lagra data i en address struct
  */

}

/*      Denna funktion ersätter det sista tecknet i strängen med '\0'.
        Använd för att ta bort '\n'.  */

int stripEnd(char *string) {
        string[strlen(string)-1] = '\0';
        return EXIT_SUCCESS;
}
```

```
/* läs in data från binära filen
   och returnera antalet strukturer */

int lasDataFilen(unsigned char * dataBuffer2, FILE *file) {

        /* Läs in längden på filen – fseek(), ftell() */
        /* Sätt tillbaka pekaren till början av filen */

        /* Beräkna antalet strukturer i filen */

        /* Allokera minne om antalet strukturer > 0 */

        /* Läs in strukturerna från filen */

        /* returnera antalet strukturer */
}

/* Skriv ut namnen på personerna i filen */

int skrivUtAdresser(unsigned char *dataBuffer2, int antalFalt) {

        /* Loopa igenom alla structurer och skriv ut namnen */

}
```