PGR3401 _H22_ 1061

Har en make file i ytterste mappe til å bygge og rengjøre alle undermapper. Jeg har også lagt debuggeren i en egen mappe som bygges som en debugger.a fil og brukes i alle oppgavene. Det blir bygget i oppgavens make fil og kopiert over til riktig mappe, inkluderer også header filen fra debuggeren ved bruk av «-I$(DEBUGFOLDER)».

Oppgave 1

a. Forklar hva C programmeringsspråket kan brukes til.

C programmeringsspråket er et generelt språk som kan brukes til masse forskjellig. Det det blir brukt til mest er programmer som må kjøre raskt og vil ikke ha en «garbage collector»/ ansvar for sitt eget minne. C blir også brukt til mikro kontrollere og IoT i dag, samt kjerne moduler i operativ systemer eller kompilatorer som må kjøre kjapt. Det blir også brukt til å kode vinruser og antivirus programmer i.

b. Hvem er Linus Torvalds og hva er han kjent for innen Informasjonsteknologi?

Linus Torvalds er en finsk programmerer som var den personen, som er kjent for å lage Linux kornellen som er førte til utviklingen av GNU/Linux operativsystemet, som mesteparten av servere bruker nå i dag. Han er også kjent for å ha laget Git som er et versionskontrollprogram som er det mest brukte versionskontrollprogram i verden.

c. Forklar hva verktøyet valgrind brukes til når man programmerer på Linux.

Valgrind er et program som brukes for å finne minnelekkasjer i programmet ditt. Man bruker dette for å finne feil og måten man kjører et program med valgrind er å kalle på valgrind i terminalen, og passere navnet på programmet du vil valgrind skal sjekke for deg (« valgrind ./main »).

Oppgave 2

Oppgave 2 fungerer, dataen ligger i data.txt og besvarelsen ligger i data_result.txt.

Tankeprosessen min var å lese en karakter om gangen og skrive den om til heksadesimal ved hjelp av %02X og rett in i resultat filen.

Oppgave 3

Oppgave 3 fungerer og har all ønsket logikk.

Dato parametere har formatet YYYYMMDD.

Punkt nummer 3 tolket jeg eldre som lægre tilbake i tid i forhold til dagens dato.

Punkt nummer 5 tolket jeg som å summere alle bookingene som har starts datoen som matcher datoen som passeres inn, og summerer opp total prisen av alle de bookingene totalt.

Oppgave 4

Har lag til funksjonaliteten med http versjon og formatert koden til å være mer lesbar også har jeg fikset de 3 feilene jeg fant.

1. Første feil jeg fant var at «MYHTTP* pHttp = (MYHTTP*)malloc(sizeof(PMYHTTP));» på linje 2 inne i funksjonen, allokerte feil antall bytes i forhold til hvor mange bytes structen trenger for all sin data inn og ikke skrive over andre sin data eller få en segmentasjons feil. Grunnen til detter er fordi du typedefer _MYHTTP til å være både MYHTTP og*PMYHTTP, det fører til at PMYHTTP er bare 8 bytes og ikke 44 som MYHTTP den trenger.

2. Andre feilen er at «pHttp->iContentLength = '0' + atoi(pszPtr);» feil kalkulerer iContentLength med 48, og det er fordi du sumerer med en char av ascii tegnet 0, som har verdien 48, så ved å fjerne den logikken så vil iContentLength fungere riktig.

3. Hadde problemer med logikken som var der angående «strchr(pszPtr, '\n')[0] = 0;» fil segmentasjon feil på dette så endret det til å finne hvor ‘\n’ er og bare bruke «strncpy» for å kopiere verdien som trengtes, fikk også satt en sperre på antall karakterer som kan gå inn i buffere for å ikke skrive over noe data.

Oppgave 5

Fungerer som det skal.

Har laget 2 pthreds, en som leser PDF filen 4096 bytes om gangen, og setter det på buffere og en som leser det og summerer det opp i et long array [256].

Har brukt flagg for å hindre at en leser og en annen skriver in i delte buffere.

Trå nummer to leser dataen og konverterer byte til en long for å så kategorisere det inn i riktig heks verdi. Så printer den ut antall av vær type av heks verdiene når done flagget er satt og den har lest all dataen.

Oppgave 6

Laget en make fil i mappen oppgave_6 for å bygge de to submappene.

Serveren og klienten er funger som det skal.

Serveren kobler seg på port 8080 og gjør seg klar for at klienten skal koble seg til og sende http GET metoder. Da leser den ut fil navnet som klienten har lyst på, prøver å lese innholdet av den og sender det til klienten for å så stenge koblingen. Hvis den ikke finner filen, sender den en 404-feilkode til klienten og lokker koblingen. Hvis klienten prøver å kalle på et annen http-metoden en GET sender den en 405-feilkode og lukker koblingen, og hvis den ikke får noe melding med data så sender den 500-feilkode og lukker koblingen. Den leser filer relativt til hvor applikasjonen ligger. Serveren fungerer også fra nettleseren.

Klienten sjekker at den har fått en parameter med navnet på filen den har lyst på, og lager en sokket og kobler seg til serveren på port 8080. da sender klienten en enkel http GET metode med filnavnet i Url-en, og får tilbake en http-respons, da printer den ut hele responsen med headers og lukker programmet.

Oppgave 7

Koden fungerer som den skal etter oppgaven.

Har ikke håndtert indetasjon til while løkkene.

Men den endrer koden til å bli while løkker og håndterer while løkke med krøllparenteser, med et ett semikolon rett etterpå og med bare en statment rett etter på.

Har inkludert en test fil til å teste de tre situasjonene og for testing av tabulator til tre mellomrom.