Afleveringsopgave 5

Afleveres senest:
Søndag d. 8. maj kl. 23.59

Jørgen Villadsen

De viste filer findes samlet i en zip-fil.

Opgaverne er ikke lige omfattende, og nogle af opgaverne kræver kendskab til emnerne, som gennemgås 19/4 og 26/4, men det er en god ide at kigge lidt på opgaverne inden da... :-)

Opgave 1

I denne opgave skal der udvikles et ret simpelt program `letters`, der tæller antallet af tegn angivet på kommandolinjen som argumenter. For eksempel (her er > operativsystemets prompt):

```
>letters abc de f gh
8
```

Tip: Benyt funktionen `strlen` erklæret i header-filen `string.h` til at bestemme længden af en streng.

Det er i orden at bruge Eclipse i stedet for kommandolinjen — i så fald skal argumenterne (`abc de f gh`) angives i menuen Run / Run Configurations og så i fanebladet Arguments.

Benyt denne fil som udgangspunkt:

```
// letters.c

#include <stdio.h>
#include <string.h>

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

// ...

  return 0;
}
```

Programmet skal testes grundigt, og dokumentationen herfor skal indgå i besvarelsen.

Opgave 2

I denne opgave skal der udvikles et program til håndtering af stakke med heltal i stil med eksemplet i lærebogens kapitel 6. Der skal dog benyttes et array i stedet for en hægtet eller lænket liste (eng. linked list).

For eksempel:

```
// stackMain.c

#include <stdio.h>
#include "stack.h"

/*

Udskrift:

popped: 4444
popped: 99
popped: 123

*/

int main() {
  stack_t * myStack = newStack();
  push(myStack, 123);
  push(myStack, 99);
  push(myStack, 4444);
  while (!empty(myStack)) {
    int value;
    value = pop(myStack);
    printf("popped: %d\n", value);
  }
  return 0;
}
```

Heltallene på stakken skal gemmes i et array. Til start skal arrayet have længden 1. Når arrayet ikke har plads til et heltal, der skal lægges på stakken (eng. push), så allokeres et nyt array i dobbelt størrelse, og heltallene kopieres. Der sker ikke ændringer af arrayets størrelse, når heltallene tages af stakken (eng. pop).

Fortsættes...

Følgende header-fil skal benyttes:

```
// stack.h

/*

Der må ikke ændres i denne header-fil.

*/

typedef struct {
  int capacity;
  int * array;
  int size;
} stack_t;

int pop(stack_t * stack_p);
void push(stack_t * stack_p, int value);
int top(stack_t * stack_p);
stack_t * newStack(void);
int empty(stack_t * stack_p);
```

Her er `capacity` længden af arrayet, og `size` er antallet af heltal i stakken.

Følgende skabelon til programmet skal benyttes:

```
// stack.c

#include <stdlib.h>
#include <stdio.h>
#include "stack.h"

/*

Tilføj funktionerne newStack, pop, push, top og empty.

*/
```

Programmet skal testes grundigt, og dokumentationen herfor skal indgå i besvarelsen.

Opgave 3

I denne opgave skal der udvikles et program kaldet `cudb`, der fungerer som en lille database over studerende, herunder deres karaktergennemsnit (eng. GPA = Grade Point Average), således som følgende session viser.

```
Welcome to CUDB - The C University Data Base

0: Halt
1: List all students
2: Add a new student

Enter action:
2

Enter name (4 characters only):
John

Enter start year (2009-2040):
2009

Enter start semester (0=Autumn/1=Spring):
0

Enter GPA (0-255):
200

Enter action:
2

Enter name (4 characters only):
Mary

Enter start year (2009-2040):
2040

Enter start semester (0=Autumn/1=Spring):
1

Enter GPA (0-255):
250

Enter action:
1

s0000 John 2009 Autumn 200
s0001 Mary 2040 Spring 250

Average GPA = 225.00

Enter action:
0

Bye
```

Fortsættes...

Linjerne efter "Enter..." er brugerens svar. Detaljerne i systemets udskrifter er ikke vigtige, så længe meningen er den samme, men der skal tages udgangspunkt i denne fil:

```
// cudb.c

#include <stdio.h>

#define NAME_SIZE 5

typedef struct {
  char name[NAME_SIZE];
  int data;
} student_t;

int main() {
  puts("Welcome to CUDB - The C University Data Base");

/*

Tilføj funktioner og anden kode efter behov.
Strukturen student_t må dog ikke ændres.

*/

  return 0;
}
```

Det specielle ved programmet er brugen af binære tal til kompakt lagring af informationer i feltet `data` i `student_t` strukturen.

Det følgende afsnit giver en kort forklaring på begrebet *bit*.

The usual decimal system is a base 10 positional system with digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9; hence it has positions for ones, tens, hundreds, thousands, etc.

$$1965 = 1 \cdot 10^3 + 9 \cdot 10^2 + 6 \cdot 10^1 + 5 \cdot 10^0 = 1000 + 900 + 60 + 5$$

Although the decimal system is the most common numeral system, some cultures have used other bases like 12, 20, and 60 for numbers.

Modern digital computers use binary numbers, that is, base 2, since the binary digits 0 and 1, also called *bits*, correspond to a current being either off or on.

$$\begin{aligned} 01010110_2 &= 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \\ &= 64 + 16 + 4 + 2 \\ &= 86 \end{aligned}$$

Here the index 2 indicates base 2.

Fortsættes...

C har avancerede faciliteter til binære tal, men i denne opgave er det tilstrækkeligt at benytte de sædvanlige aritmetiske operatorer på heltal `+-*/` og `%` (rest ved division).

Betragt dette lille program:

```
// bits.c

#include <stdio.h>

/*

Udskrift:

86 = 01010110_2

*/

int main() {
  int d = 86;
  int b0 = d % 2;
  int b1 = (d / 2) % 2;
  int b2 = (d / 4) % 2;
  int b3 = (d / 8) % 2;
  int b4 = (d / 16) % 2;
  int b5 = (d / 32) % 2;
  int b6 = (d / 64) % 2;
  int b7 = (d / 128) % 2;
  printf("%d = %d%d%d%d%d%d%d%d_2",
      d, b7, b6, b5, b4, b3, b2, b1, b0);
  return 0;
}
```

Bemærk hvordan de 8 bit betegnes 0–7 "fra højre mod venstre" (svarende til 0 for den mindst betydende bit).

Feltet `data` i `student_t` strukturen skal benyttes til at lagre informationerne om de enkelte studerende som følger:

> Bit 0–4: Startår (med udgangspunkt i år 2009, altså for eksempel 1 betyder 2010).
>
> Bit 5: Startsemester (0 for efterår og 1 for forår).
>
> Bit 6–13: Karaktersnit (255 er naturligvis den bedste karakter).

Bit 14 og højere skal ikke benyttes i denne opgave og kan blot sættes til 0.

Der skal være plads 10000 studerende i databasen.

Programmet skal testes grundigt, og dokumentationen herfor skal indgå i besvarelsen.