Programmering uge 7

Opgave 1a
Sine test
Sine function

```c
#include "taylor_sine.h"
#include <stdio.h>

double factorial(int n)
{
    if (n == 0 || n == 1)
    {
        return 1.0;
    }
    double fact = 1.0;

    for (int i = 2; i <= n; i++)
    {
        fact *= i;
    }
    return fact;
}

double power(double tal, int potens)
{
    double resultat = 1.0;
    for (int i = 0; i < potens; i++)
    {
        resultat *= tal;
    }
    return resultat;
}

double taylor_sine(double x, int n)
{
    double sine = 0.0;
    for (int i = 0; i < n; i++)
    {
        double exponent = 2 * i + 1;
        double n = power(x, exponent) / factorial(exponent);
        if (i % 2 == 0)
        {
            sine += n;
        }
        else
        {
            sine -= n;
        }
```

```
    }
    return sine;
}
```

Opgave 1b

```
#include <stdio.h>
#include "taylor_sine.h"
#include <math.h>

#define pi 3.1415926535


int main(){
    double x_values[]={0, pi/2, pi, 2*pi, pi/6, 100};
    int n = 10;

    for (int i=0; i<6; i++){
        double x = x_values[i];
        double approks_sine = taylor_sine(x, n);
        double math_sine = sin(x);

        printf("\n");
        printf("x = %lf\n", x);
        printf("Taylorrækkens sinus: %lf\n", approks_sine);
        printf("Matematik funktionens sinus: %lf\n", math_sine);
        printf("Differensen: %lf\n", fabs(approks_sine - math_sine));
    }
    printf("\n");

    return 0.0;
}
```

Test- cases:

x = 0.000000
Taylorrækkens sinus: 0.000000
Matematik funktionens sinus: 0.000000
Differensen: 0.000000

x = 1.570796
Taylorrækkens sinus: 1.000000
Matematik funktionens sinus: 1.000000
Differensen: 0.000000

x = 3.141593
Taylorrækkens sinus: -0.000000
Matematik funktionens sinus: 0.000000
Differensen: 0.000000

x = 6.283185
Taylorrækkens sinus: -0.001048
Matematik funktionens sinus: -0.000000
Differensen: 0.001048

x = 0.523599
Taylorrækkens sinus: 0.500000
Matematik funktionens sinus: 0.500000
Differensen: 0.000000

x = 100.000000
Taylorrækkens sinus: -794697857233432870912.000000
Matematik funktionens sinus: -0.506366
Differensen: 794697857233432870912.000000

Opgave 1c
Stack
Test stack

```
/*
Opgave C:
Which intervals of input x did your function give a similar result to the ANSI C sin
function?
- Programmet taylor_sine giver de rigtige resultater på lave værdier for x (0 til pi),
mens det i værdier der kommer over pi er mere upræcist.

What impact did increasing the precision have (i.e. increasing the number of Taylor
series terms)?
- ved test med n=20 blev resultaterne for funktionen mere præcis.
*/
```

Opgave 2a

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

void initialize(stack *s) {
  s->head = NULL; //stack er initaliseret til at være tom
}

void push(int x, stack *s) {
  node* ny_node = (node*)malloc(sizeof(node));  // lav ny node

  ny_node->data = x;     // tildel data til noden
  ny_node->next = s->head;  // den nye node peger på head
  s->head = ny_node;     // opdater head til den nye node
}
```

```
int pop(stack *s) {
  if (empty(s)) {
    printf("Ikke flere værdier at poppe i stack.\n");
    return -1;
  }
  int værdi = s->head->data;  // få værdien fra den øverste node
  node* temp = s->head;       // gem den nuværende head
  s->head = s->head->next;    // flyt head til den næste node
  free(temp);                 // frigør den gamle head node
  return værdi;
}

bool empty(stack *s) {
  if (s->head = NULL){
    return true;
  }
  return false;
}

bool full(stack *s) {
  return false; //kan aldrig være 'full,' da det er en linked list -måske det der var
tricket?
}
```

Opgave 2b

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

int main()
{
    stack s;
    int x;
    int a;
    int b;

    //test: initialize(s) - sikrer at stack er tom
    initialize(&s);
    assert(empty(&s));

    //test: push og pop
    push(10, &s);
    x = pop(&s);
    assert(x == 10);
    assert(empty(&s));

    //test: LIFO rækkefølge
    push(2, &s);
    push(5, &s);
```

```
    a = pop(&s);
    b = pop(&s);
    assert(a == 2);
    assert(b == 5);
    assert(empty(&s));

    printf("alle tests bestod");
    return 0;
}
```