Tölvunarfræði 1 - Heimadæmi 01

Teitur Guðmundarson

1. Skrifið fall sem tekur inn tvær tölur x og y þar sem x $<$ y og skrifar út allar Fibonacci tölur F_i þar sem $x < F_i < y$. Skrifið eina tölu í hverja línu.

Fibonacci.c

```c
#include <stdio.h>
int fib(int x, int y) {
    int a = 0; int b = 1;
    for(int i = 0; i <= y; i++) {
        if (a > x && a < y)
            printf("%d\n", a);
        int sum = a + b;
        a = b;
        b = sum;
    }
}
```

2. Skrifið fallið **my_strlen** sem tekur inn streng og reiknar út lengdina á strengnum. Prufið fallið ykkar með því að bera saman við **strlen** sem er skilgreint í string.h. Tilgreiningin á fallinu er gefin hér að neðan

streng.c

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

int my_strlen(char s[]) {
    int i = 0;
    int lengd = 0;
        while (s[i] != '\0') {
            i++;
            lengd++;
        } return lengd;
}

int main() {
    char a[] = "heimadaemi";
    int my_str = my_strlen(a);
    int str = strlen(a);
    printf("my_str = %d\n", my_str);
    printf("str = %d\n", str);
}
```

3. Skrifið fall **my_stats** sem tekur inn fylki af fleytitölum og lengdina á fylkinu og skilar meðaltali, minnstu og stærstu tölu í fylkinu. Skilgreinið **struct** sem inniheldur þessar upplýsingar og prufið fallið ykkar í **main** fallinu með a.m.k. tveimur ólíkum fylkjum og prentið út á læsilegan hátt. Skilið öllum forritskóðanum og niðurstöðum úr keyrslum (ath. bara texti, engin skjáskot)

stats.c

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

struct uppl {
    double max;
    double min;
    double avg;
};

double my_stats(int length, double arr[]) {
    struct uppl a;
    a.max = arr[0];
    a.min = arr[0];
    a.avg = 0.0;
    // finna max
    for (int i = 0; i < length; i++) {
        if (arr[i] > a.max)
            a.max = arr[i];
    }
    // finna min
    for (int i = 0; i < length; i++) {
        if (arr[i] < a.min)
            a.min = arr[i];
    }
    a.avg = (a.min+a.max) / 2;
    return a.avg;
}

int main() {
    double arr[] = { 1.1, 2.2, 3.3, 4.4 };
    int length = 0;
    double arr2[] = { 3.14, 9.007, 5.728, 0.19, 7.2 };
    int length2 = 0;
    int i = 0;
    int j = 0;
        while (arr[i] != '\0') {
            i++;
            length++; // skilar 3, veit ekki afk
        } length++; // skítalausn til að fá length=4
        while (arr2[j] != '\0') {
            j++;
            length2++; // skilar réttri lengd hér!?
        }
    double prentist = my_stats(length, arr);
    printf("Fylki1: %.5f ", prentist);
    double prentist2 = my_stats(length2, arr2);
    printf("Fylki2: %.5f\n", prentist2);
}
```

Niðurstaða:
Fylki1: 2.75000 Fylki2: 4.59850

4. Byrjið með kóðann sem er gefinn í **complex.c** og bætið við falli **complex_abs** sem reiknar út lengdina á tvinntölunni $a + bi$ með formúlunni

$$| a + bi | = \sqrt{a^2 + b^2}$$

Til að reikna út kvaðratrótina notið þið **sqrt** fallið sem er skilgreint í **math.h** forritasafninu. Athugið að við þýðingu þurfið þið að bæta við **-lm**, annars finnst fallið **sqrt ekki** (hér þarf bara að skila fallinu en þið ættuð að prufa hvort það virki).

abs.c

```
double complex_abs(struct complex d) {
   double abs = sqrt(d.re*d.re + d.im*d.im);
   return abs;
}
```

5. Skrifið forrit sem tekur inn heiltölu N af viðfangi og prentar út **N is a prime number** ef hún er prímtala en **N is composite, k is the smallest prime factor** ef hún er það ekki (skiptið út gildum fyrir N og k eftir því sem við á). Ef N er ekki prímtala þá látum við k vera minnstu prímtölu sem gengur upp í N.

prime.c

```c
#include <stdio.h>
#include <stdlib.h>

int isPrime(int N) {
    int k = 0;
    for (int i = 2; i < N-1; i++) {
        if (N % i == 0) {
            k = i;
            return k;
        }
    } return 1;
}

int main(int argc, char* argv[]) {
    int n = atoi(argv[1]);
    int x = isPrime(n);
    if (x == 1) printf("%d is prime\n", n);
    else printf("%d is smallest prime factor\n", x);
}
```