Aufgabenblatt 10: Rekursion und Zeichenketten

Florian Ludewig (Übungsgruppe 2)

10. Januar 2020

Aufgabe 1 – Rekursive Algorithmen

a)

```c
#include<stdio.h>

int fibonacci(int n ) {
  int fibonacci_number = 1;
  int penultimate = 0;
  int previous = 1;
  for (int i = 1; i <= n; i++) {
    fibonacci_number = penultimate + previous;
    penultimate = previous;
    previous = fibonacci_number;
  }
  return fibonacci_number;
}

int main(void) {
  int n;
  printf("Zahl eingeben: ");
  scanf("%d", &n);
  printf("\nFibonacci number of %d is %d\n", n, fibonacci(n));
  return 0;
}
```

b)

```c
#include<stdio.h>

int count_digit(int num, int digit) {
  int counter = 0;
  while(1) {
    int last_digit = num % 10;
    num /= 10;
    if (last_digit == digit) counter++;
    if (num == 0) break;
  }
  return counter;
}

int main(void) {
  int num, digit;
  printf("Zahl eingeben: ");
  scanf("%d", &num);
  printf("Ziffer eingeben: ");
  scanf("%d", &digit);
  int x = count_digit(num, digit);
  printf("Die Ziffer %d kommt %d mal in %d vor\n", digit, x, num);
  return 0;
}
```

c)

```c
#include<stdio.h>

int is_sorted(int num) {
  int last_digit;
  int penultimate_digit;
  while(1) {
    last_digit = num % 10;
    num /= 10;
    penultimate_digit = num % 10;
    if (num == 0) break;
    if (last_digit > penultimate_digit) return 0;
  }
  return 1;
}

int main(void) {
  int num;
  printf("Zahl eingeben: ");
  scanf("%d", &num);
  int sorted = is_sorted(num);
  if (sorted)
    printf("%d ist von links nach rechts absteigend sortiert\n", num);
  else
    printf("%d ist nicht von links nach rechts absteigend sortiert\n", num);
  return 0;
}
```

Aufgabe 2 – Verschiebechiffre

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

char* encipher(char string[], int key) {
  int length = strlen(string);
  char *encoded = (char *) malloc(sizeof(char) * length);
  for (int i = 0; i < strlen(string); i++) {
    if (string[i] == ' ') encoded[i] = ' ';
    else encoded[i] = string[i] + key;
  }
  return encoded;
}

char* decipher(char string[], int key) {
  int length = strlen(string);
  char *decoded = (char *) malloc(sizeof(char) * length);
  for (int i = 0; i < strlen(string); i++) {
    if (string[i] == ' ') decoded[i] = ' ';
    else decoded[i] = string[i] - key;
  }
  return decoded;
}

int main(void) {
  char text[] = "YLHOH NDPHQ DOOPDHKOLFK CX GHU XHEHUCHXJXQJ HLQHQ JURVVHQ
      IHKOHU JHPDFKW CX KDEHQ DOV VLH YRQ GHQ EDHXPHQ KHUXQWHUJHNRPPHQ ZDUHQ
       XQG HLQLJH VDJWHQ VFKRQ GLH EDHXPH VHLHQ HLQ KROCZHJ JHZHVHQ GLH
      RCHDQH KDHWWH PDQ QLHPDOV YHUODVVHQ GXHUIHQ";
  char* decoded = decipher(text, 3);
  printf("Klartext: %s\n", decoded);
  printf("Verschluesselt: %s\n", encipher(decoded, 3));
  return 1;
}
```