Aufgabenblatt 11: Dynamische Datenstrukturen

Florian Ludewig (Übungsgruppe 2)

18. Januar 2020

Aufgabe 1 – Einfach verkettete Listen

Ich habe die Aufgabe so interpretiert, dass `insert_sorted` die Zahlen aufsteigend ordnet.

```
void insert_sorted(int val) {
  struct node *temp = head;
  while (temp -> next != NULL && temp -> next -> data < val) {
    temp = temp -> next;
  }
  struct node *inserted = makeNode(val);
  inserted -> next = temp -> next;
  temp -> next = inserted;
}

struct node *reverse() {
  struct node *previous = NULL;
  struct node *current = head;
  struct node *next = NULL;
  while (current != NULL) {
    next = current -> next;
    current -> next = previous;
    previous = current;
    current = next;
  }
  return previous;
}
```

Aufgabe 2 – Stapel

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

struct node {
  char data;
  struct node *next;
};

struct node *top;

struct node *makeNode(char val) {
  struct node *node = NULL;
  node = malloc(sizeof(struct node));
  if (node != NULL) {
    node -> data = val;
    node -> next = NULL;
    return node;
  } else {
    return NULL;
  }
}

```

```c
void push(char c){
  struct node *p = makeNode(c);
  p -> next = top;
  top = p;
}

int pop(void){
  int result = top -> data;
  struct node *p = top;
  top = top -> next;
  free(p);
  return result;
}

int is_opening_bracket(char c) { return c == '{' || c == '[' || c == '('; }
int is_closing_bracket(char c) { return c == ')' || c == ']' || c == '}'; }
int is_matching_bracket(char opening, char closing) {
  if (opening == '(') return closing == ')';
  if (opening == '[') return closing == ']';
  if (opening == '{') return closing == '}';
  return 0;
}

void print_error(char input[], int position) {
    printf("Fehlerhaft Klammerung:\n");
    printf("%s\n", input);
    for (int i = 0; i < position; i++) {
      printf(" ");
    }
    printf("^\n");
}

void validate_brackets(char input[]) {
  size_t length = strlen(input);
  for (int i = 0; i < length; i++) {
    char c = input[i];
    if (is_opening_bracket(c)) {
      push(c);
    }
    if (is_closing_bracket(c)) {
      if (is_matching_bracket(top -> data, c)) {
        pop();
      } else {
        print_error(input, i);
        return;
      }
    }
  }
  if (!top) {
    printf("Korrekte Klammerung\n");
  } else {
    print_error(input, length);
  }
}

int main(void) {
  char input[256];
  printf("Bitte Ausdruck eingeben: ");
  scanf("%s", &input[0]);
  validate_brackets(input);
  return 1;
}
```