Aufgabenblatt 12: Dynamische Datenstrukturen (2)

Florian Ludewig (Übungsgruppe 2)

24. Januar 2020

Aufgabe 1 – Doppelt verkettete Listen

a), b)

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

struct dnode *head, *last;

struct dnode{
  int data;
  struct dnode *next, *prev;
};

struct dnode *mkNode(int val){
  struct dnode *node = NULL;
  if((node = malloc(sizeof (struct dnode))) != NULL) {
    node -> data = val;
    node -> next = node -> prev = NULL;
    return node;
  }
  else return NULL;
}

void printList(void) {
  if (head == NULL) {
    printf("( )");
    return;
  }
  printf("( ");
  struct dnode *tmp = head;
  while(tmp != NULL){
  printf("%d ", tmp -> data);
    tmp = tmp -> next;
  }
  printf(")\n");
}

struct dnode *insert_start(int val) {
  struct dnode *new_node = mkNode(val);
  if (head == NULL) {
    new_node -> next = NULL;
    head = new_node;
    last = head;
  } else {
    new_node -> next = head;
    head -> prev = new_node;
    head = new_node;
  }
  new_node -> prev = NULL;
  return new_node;
}
```

```c

void remove_element(int val) {
  if (head -> next == NULL) return;
  struct dnode *deleted;
  if (head -> data == val) {
    deleted = head;
    head = head -> next;
    head -> prev = NULL;
  } else {
    struct dnode *temp = head;
    while(temp -> data != val && temp -> next != NULL) {
      temp = temp -> next;
    }
    if (temp == NULL) return;
    deleted = temp;
    (temp -> prev) -> next = temp -> next;
    if (temp -> next != NULL) {
      temp -> next -> prev = temp -> prev;
    }
  }
  if (deleted) free(deleted);
}

int main(void) {
  int remove, primes[] = { 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41,
     43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97 };
  for (int i = sizeof(primes) / sizeof(int) - 1; i >= 0; i--) {
    insert_start(primes[i]);
  }
  printf("Zahl eingeben: ");
  scanf("%d", &remove);
  remove_element(remove);
  printList();
  return 1;
}
```

b)

Aufgabe 2 – Binärbäume

```
...
```