**Program Studi Teknik Elektro ITB** Nama Kuliah (Kode) : Praktikum Pemecahan Masalah dengan C (EL2208)

Tahun / Semester : 2023-2024 / Genap

**Modul** : 4 – Recursion

**Nama Asisten / NIM** : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Nama Praktikan / NIM** : Pradigta Hisyam Ramadhan / 18322008

**Tugas Pendahuluan**

|  |
| --- |
| **Soal Teori**  1. Apabila suatu implementasi rekursif **tidak memiliki kasus basis** (*base case*), maka pemanggilan fungsi/program rekursif akan selalu dipanggil dan tidak pernah berhenti. Akibatnya, program akan terjebak di dalam *infinite loop*.  2. Salah satu kasus yang dapat diselesaikan dengan rekursif adalah **pemrosesan dalam sebuah *array* atau *list* sederhana**. Namun, pemecahan masalah tersebut dengan rekursif tidak disarankan karena kompleksitasnya yang terlampau rumit dan membutuhkan waktu eksekusi yang lebih lama. Oleh karena itu, pencarian dalam *array* atau *list* lebih sering menggunakan pendekatan iteratif karena lebih efisien dan mudah untuk dipahami.  3. Implementasi program secara **iteratif** akan memiliki proses eksekusi yang **lebih cepat** dibandingkan dengan implementasi menggunakan pendekatan secara rekursif. Hal ini disebabkan oleh perbedaan **kompleksitas waktu (time complexity)** pada kedua pendekatan. Pada algoritma rekursif, kompleksitas waktu dapat meningkat secara eksponensial karena fungsi rekursif akan terus dipanggil sampai kasus basis tercapai. Kompleksitas waktu dari sebuah program rekursif dapat ditentukan dengan menghitung seberapa banyak fungsi tersebut dipanggil oleh dirinya sendiri (hitung n-kali fungsi dipanggil). |
| **Soal Pemrograman**  /\*EL2208 Praktikum Pemecahan Masalah dengan C 2023/2024  \*TP Modul         : 4 - Recursion  \*Hari dan Tanggal : Selasa, 19 Maret 2024  \*Nama (NIM)       : Pradigta Hisyam Ramadhan (18322008)  \*Nama File        : tp04-soalPemrograman.c  \*Deskripsi        : Program digunakan untuk mencari jumlah pengulangan sebuah bilangan, apabila                      kuadrat-kuadrat digitnya dikuadratkan sampai mendapatkan hasil 1.  \*/  #include <stdio.h>  #define MAX\_SIZE 1000  // Fungsi rekursif untuk menghitung sequence angka  int countIteration(int n, int visited[MAX\_SIZE]) {      // Deklarasi variabel lokal      int sum, temp, digit;      // Kasus basis      if (n == 1){          return 0;      }      else if (visited[n] == 1){          // Angka diulang dua kali, menuju infinite loop, diindikasi dengan nilai -1 (error)          return -1;      }      else{          // Beri penanda pada array untuk angka yang sudah dilewati          visited[n] = 1;          // Menjumlahkan kuadrat digit-digit angka yang diinput          sum = 0;          temp = n;          while(temp != 0){              digit = temp % 10;              sum += digit \* digit;              temp /= 10;          }          // Rekursif          int nextN = countIteration(sum, visited);          if (nextN == -1){              return -1; // exit infinite loop          }          return 1 + nextN;      }  }  void checkFiniteLoop(int iteration){      if (iteration == -1){          printf("Infinite.");      }      else{          printf("%d", iteration);      }  }  int main() {      int number, pengulangan;      int visited[MAX\_SIZE]; // Array untuk menyimpan angka yang sudah dikunjungi      // Mendapatkan bilangan      printf("Masukkan bilangan N: ");      scanf("%d", &number);        // Memanggil fungsi rekursif untuk menghitung sequence angka dan mencetak hasilnya      pengulangan = countIteration(number, visited);      // Cek apakah pengulangan tak berhingga atau tidak      checkFiniteLoop(pengulangan);      return 0;  }  **Output yang dihasilkan pada terminal** |

**Referensi**

*4.8. Infinite Recursion — How to Think Like a Computer Scientist - C++*. (n.d.). Retrieved March 18, 2024, from <https://runestone.academy/ns/books/published/thinkcpp/Chapter4/InfiniteRecursion.html>

G. (2023, May 22). *Difference between Recursion and Iteration*. GeeksforGeeks. Retrieved March 18, 2024, from <https://www.geeksforgeeks.org/difference-between-recursion-and-iteration/>