```
Setiawan Muhammad
1203230016
IF 03 01

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

// Struktur Node untuk menyimpan huruf dan pointer ke Node berikutnya
struct Node {
    char* alphabet; // Menyimpan karakter huruf
    struct Node* link; // Pointer ke Node berikutnya
};

int main() {
    // Deklarasi node-node
    struct Node l1, l2, l3, l4, l5, l6, l7, l8, l9;
    struct Node *link, *l3ptr;

    // Inisialisasi node-node dengan menggunakan potongan kode soal
    l1.link = NULL;
    l1.alphabet = "F";

    l2.link = NULL;
    l2.alphabet = "M";

    l3.link = NULL;
    l3.alphabet = "A";

    l4.link = NULL;
    l4.alphabet = "I";

    l5.link = NULL;
    l5.alphabet = "K";
```

```
    l6.link = NULL;
    l6.alphabet = "T";

    l7.link = NULL;
    l7.alphabet = "N";

    l8.link = NULL;
    l8.alphabet = "O";

    l9.link = NULL;
    l9.alphabet = "R";

    // Mengatur koneksi antar node sesuai dengan urutan yang diinginkan
    l7.link = &l1;// Menyambungkan ke l1
    l1.link = &l8;// Menyambungkan ke l8
    l8.link = &l2;// Menyambungkan ke l2
    l2.link = &l5;// Menyambungkan ke l5
    l5.link = &l3;// Menyambungkan ke l3
    l3.link = &l6;// Menyambungkan ke l6
    l6.link = &l9;// Menyambungkan ke l9
    l9.link = &l4;// Menyambungkan ke l4
    l4.link = &l7;// Menyambungkan ke l7

    // Starting point
    l3ptr = &l7;

    // Akses data menggunakan printf
    printf("%s", l3.link->link->link->alphabet);// Menampilkan huruf I
    printf("%s", l3.link->link->link->link->alphabet);// Menampilkan huruf N
    printf("%s", l3.link->link->link->link->link->alphabet);// Menampilkan huruf F
    printf("%s", l3.link->link->link->link->link->link->alphabet);// Menampilkan
huruf O
    printf("%s", l3.link->link->alphabet);// Menampilkan huruf R
```

```
    printf("%s", l3.link->link->link->link->link->link->link->alphabet);//
Menampilkan huruf M
    printf("%s", l3.alphabet);// Menampilkan huruf A
    printf("%s", l3.link->alphabet);// Menampilkan huruf T
    printf("%s", l3.link->link->link->alphabet);// Menampilkan huruf I
    printf("%s", l3.link->link->link->link->link->link->link->link->alphabet);//
Menampilkan huruf K
    printf("%s", l3.alphabet);// Menampilkan huruf A


    return 0;
}
```

```
70
71      return 0;
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

INFORMATIKA
PS D:\Algoritma Struktur Data>
```

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

// Deklarasi fungsi-fungsi yang akan digunakan
char* readline();
```

```c
char* ltrim(char*);
char* rtrim(char*);
char** split_string(char*);
int parse_int(char*);
int twoStacks(int maxSum, int a_count, int* a, int b_count, int* b);

// Fungsi utama
int main() {
    // Deklarasi dan inisialisasi file pointer untuk output
    FILE* fptr = fopen(getenv("OUTPUT_PATH"), "w");

    // Membaca jumlah kasus dari input dan mengonversinya ke dalam integer
    int g = parse_int(ltrim(rtrim(readline())));

    // Loop untuk setiap kasus
    for (int g_itr = 0; g_itr < g; g_itr++) {
        // Memecah input untuk setiap kasus dan mengonversi menjadi
integer
        char** first_multiple_input = split_string(rtrim(readline()));
        int n = parse_int(*(first_multiple_input + 0));
        int m = parse_int(*(first_multiple_input + 1));
        int maxSum = parse_int(*(first_multiple_input + 2));

        // Memecah input untuk nilai-nilai tumpukan A
        char** a_temp = split_string(rtrim(readline()));
        int* a = malloc(n * sizeof(int));
        for (int i = 0; i < n; i++) {
            int a_item = parse_int(*(a_temp + i));
            *(a + i) = a_item;
        }

        // Memecah input untuk nilai-nilai tumpukan B
        char** b_temp = split_string(rtrim(readline()));
        int* b = malloc(m * sizeof(int));
```

```c
        for (int i = 0; i < m; i++) {
            int b_item = parse_int(*(b_temp + i));
            *(b + i) = b_item;
        }

        // Memanggil fungsi twoStacks untuk menyelesaikan kasus ini
        int result = twoStacks(maxSum, n, a, m, b);

        // Menulis hasil ke file output
        fprintf(fptr, "%d\n", result);

        // Membebaskan memori yang dialokasikan untuk array a dan b
        free(a);
        free(b);
    }

    // Menutup file output
    fclose(fptr);

    return 0;
}

// Fungsi-fungsi bantu untuk membaca, memproses, dan memecah string
char* readline() {
    // Implementasi fungsi readline()
}

char* ltrim(char* str) {
    // Implementasi fungsi ltrim()
}

char* rtrim(char* str) {
    // Implementasi fungsi rtrim()
}
```

```c
char** split_string(char* str) {
    // Implementasi fungsi split_string()
}

int parse_int(char* str) {
    // Implementasi fungsi parse_int()
}

// Fungsi utama untuk menyelesaikan masalah dua tumpukan
int twoStacks(int maxSum, int a_count, int* a, int b_count, int* b) {
    // Implementasi fungsi twoStacks()
}
```