

Benediktus Daniel Afriant

1203230005

## 1. Asisten Sherlock Holmes

### Source Code

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

struct Node {
    char* alphabet;
    struct Node* link;
};

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";
```

```

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

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

// Mengatur koneksi antar node sesuai dengan urutan yang diinginkan
17.link = &l1; // Menyambungkan ke l1
11.link = &l8; // Menyambungkan ke l1
18.link = &l2; // Menyambungkan ke l1
12.link = &l5; // Menyambungkan ke l1
15.link = &l3; // Menyambungkan ke l1
13.link = &l6; // Menyambungkan ke l1
16.link = &l9;
19.link = &l4;
14.link = &l7;

// Starting point
l3ptr = &l7;

// Akses data menggunakan printf
printf("%s", 13.link->link->link->alphabet); // Menampilkan huruf I
printf("%s", 13.link->link->link->link->alphabet); // Menampilkan huruf N
printf("%s", 13.link->link->link->link->link->alphabet); // Menampilkan
huruf F
printf("%s", 13.link->link->link->link->link->link->alphabet); //
Menampilkan huruf O
printf("%s", 13.link->link->alphabet); // Menampilkan huruf R
printf("%s", 13.link->link->link->link->link->link->link->alphabet); //
Menampilkan huruf M
printf("%s", 13.alphabet); // Menampilkan huruf A
printf("%s", 13.link->alphabet); // Menampilkan huruf T
printf("%s", 13.link->link->link->alphabet); // Menampilkan huruf I
printf("%s", 13.link->link->link->link->link->link->link->link->
>alphabet); // Menampilkan huruf K
printf("%s", 13.alphabet); // Menampilkan huruf A

return 0;
}

```

Output :

```
PS C:\Users\Daniel\Downloads\PS1\output> & .\'Test 10.exe'  
INFORMATIKA  
PS C:\Users\Daniel\Downloads\PS1\output> █
```

### Penjelasan

`#include <stdio.h>`: Mengimpor library standar untuk input-output dalam bahasa C.

`#include <stdlib.h>`: Mengimpor library standar untuk fungsi-fungsi umum seperti alokasi memori dinamis.

`struct Node`: Mendefinisikan sebuah struktur bernama Node untuk menyimpan data dan pointer ke simpul berikutnya.

`char* alphabet`:: Variabel dalam struktur Node untuk menyimpan huruf.

`struct Node* link`:: Pointer dalam struktur Node untuk menunjuk ke simpul berikutnya.

`int main() {`: Fungsi utama program.

`struct Node l1, l2, l3, l4, l5, l6, l7, l8, l9`:: Mendeklarasikan sembilan simpul Node sebagai variabel lokal.

`struct Node *link, *l3ptr`:: Mendeklarasikan pointer yang akan digunakan dalam program.

Pendeklarasian dan inisialisasi simpul-simpul dengan data huruf yang diinginkan.

Pengaturan koneksi antar simpul sesuai dengan urutan yang diinginkan, membentuk sebuah lingkaran.

`l3ptr = &l7`:: Menetapkan pointer l3ptr untuk menunjuk ke simpul l7 sebagai

titik awal.

Penggunaan printf untuk mengakses data dalam linked list dan mencetak huruf-huruf yang tersimpan dalam simpul-simpul yang terhubung.

return 0; Mengembalikan nilai 0, menandakan bahwa program selesai dijalankan dengan sukses.

## 2. Soal Hackerrank

### Source Code

```
#include <stdio.h>

// Fungsi untuk mencari nilai maksimum dari dua bilangan
int max(int a, int b) {
    return a > b ? a : b;
}

// Fungsi untuk mencari jumlah maksimum elemen yang dapat diambil
// dari kedua stack
int twoStacks(int maxSum, int a[], int m, int b[], int n) {
    int total = 0, count = 0, maxCount = 0;
    int i = 0, j = 0;

    // Iterasi untuk mengambil elemen dari kedua stack
    while (i < m && total + a[i] <= maxSum) {
        total += a[i++];
        count++;
        maxCount = max(maxCount, count);
    }

    // Iterasi sambil mempertimbangkan elemen dari stack kedua
    while (j < n && i >= 0) {
        total += b[j++];
        count++;

        // Jika total melebihi maxSum, kurangi elemen dari stack
        // pertama
        while (total > maxSum && i > 0) {
            i--;
            total -= a[i];
            count--;
        }
    }
}
```

```

    }

    // Perbarui maxCount jika jumlah saat ini lebih besar
    if (total <= maxSum) {
        maxCount = max(maxCount, count);
    }
}

return maxCount;
}

int main() {
    int games;
    scanf("%d", &games);

    for (int g = 0; g < games; g++) {
        int m, n, maxSum;
        scanf("%d %d %d", &m, &n, &maxSum);
        int a[m], b[n];

        // Masukkan elemen stack pertama
        for (int i = 0; i < m; i++) {
            scanf("%d", &a[i]);
        }

        // Masukkan elemen stack kedua
        for (int i = 0; i < n; i++) {
            scanf("%d", &b[i]);
        }

        // Cetak hasil jumlah maksimum elemen yang dapat diambil
        printf("%d\n", twoStacks(maxSum, a, m, b, n));
    }

    return 0;
}

```

Output

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

### ✓ Sample Test case 0

Input (stdin)

[Download](#)

```
1 1
2 5 4 10
3 4 2 4 6 1
4 2 1 8 5
```

Your Output (stdout)

```
1 4
```

Expected Output

[Download](#)

```
1 4
```

## Penjelasan

1. `#include <stdio.h>`: Mengimpor library standar untuk input-output dalam bahasa C.
2. `int max(int a, int b) {` : Deklarasi fungsi `max` yang mengambil dua nilai integer dan mengembalikan nilai yang lebih besar di antara keduanya.
3. `return a > b ? a : b;` : Jika `a` lebih besar dari `b`, maka fungsi `max` akan mengembalikan nilai `a`. Jika `b` lebih besar dari `a`, maka fungsi `max` akan mengembalikan nilai `b`. Jika `a` dan `b` sama-sama besar, maka fungsi `max` akan mengembalikan salah satu nilai.
4. `int twoStacks(int maxSum, int a[], int m, int b[], int n) {` :Deklarasi fungsi `twoStacks` yang mengambil jumlah maksimum (`maxSum`), dua array integer (`a[]` dan `b[]`), serta panjang array `m` dan `n`. Fungsi ini mengembalikan jumlah maksimum elemen yang dapat diambil dari kedua stack.
5. `int total = 0, count = 0, maxCount = 0;` :Pendefinisian variabel-variabel lokal seperti `total` untuk menyimpan total nilai yang diambil dari stack, `count` untuk menghitung jumlah elemen yang telah diambil, dan `maxCount` untuk menyimpan jumlah maksimum elemen yang dapat diambil.
6. `while (i < m && total + a[i] <= maxSum) {` : Iterasi pertama untuk mengambil elemen dari stack pertama (`a[]`). Jika penambahan elemen ke `total` masih belum melebihi `maxSum`, elemen tersebut diambil dan `count` serta `maxCount` diperbarui jika perlu.
7. `while (j < n && i >= 0) {` : Iterasi kedua untuk mengambil elemen dari stack kedua (`b[]`), sambil mempertimbangkan jumlah total yang telah diambil dari stack pertama.
8. `return maxCount;` Fungsi `twoStacks` mengembalikan nilai `maxCount`, yang merupakan jumlah maksimum elemen yang dapat diambil dari kedua stack tanpa melebihi `maxSum`. (Baris ke-42)
9. `int main() {` :Fungsi `main()` sebagai titik awal eksekusi program.
10. `scanf("%d", &games);` : Membaca jumlah permainan dari input

11. `for (int i = 0; i < m; i++) {` : Untuk setiap permainan, membaca jumlah elemen dari stack pertama (m), stack kedua (n), dan maxSum.
12. `return 0;` : Program selesai dan mengembalikan nilai 0.