

NAMA : MUH. AGGUM NIAS PUTRA

NIM : 1203230047

KELAS : IF-03-01

TUGAS PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

Input :

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

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

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

```
int main() {
```

```
    struct Node l1, l2, l3, l4, l5, l6, l7, l8, l9;
```

```
    struct Node *link, *l3ptr;
```

```
    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";

l7.link = &l1;

l1.link = &l8;

l8.link = &l2;

l2.link = &l5;

l5.link = &l3;

l3.link = &l6;

l6.link = &l9;

l9.link = &l4;

l4.link = &l7;

l3ptr = &l7;

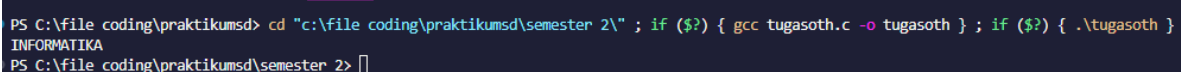
```

printf("%s", l3.link->link->link->alphabet);
printf("%s", l3.link->link->link->link->alphabet);
printf("%s", l3.link->link->link->link->link->alphabet);
printf("%s", l3.link->link->link->link->link->link->alphabet);
printf("%s", l3.link->link->alphabet);
printf("%s", l3.link->link->link->link->link->link->link->alphabet);
printf("%s", l3.alphabet);
printf("%s", l3.link->alphabet);
printf("%s", l3.link->link->link->alphabet);
printf("%s", l3.link->link->link->link->link->link->link->link->alphabet);
printf("%s", l3.alphabet);

return 0;
}

```

Output :



```

PS C:\file coding\praktikumsd> cd "c:\file coding\praktikumsd\semester 2\" ; if ($?) { gcc tugasoth.c -o tugasoth } ; if ($?) { .\tugasoth }
INFORMATIKA
PS C:\file coding\praktikumsd\semester 2>

```

Penjelasan :

1. `#include <stdio.h>`: Menggunakan direktif preprosesor `#include` untuk memasukkan file header `stdio.h` yang menyediakan fungsi input-output standar.
2. `#include <stdlib.h>`: Menggunakan direktif preprosesor `#include` untuk memasukkan file header `stdlib.h` yang menyediakan fungsi-fungsi umum, seperti alokasi memori dinamis, konversi tipe data, dll.
3. `struct Node { ... };`: Mendefinisikan struktur `Node` yang berisi pointer ke tipe data `char` dan pointer ke struktur `Node` itu sendiri.
4. `int main() { ... }`: Fungsi utama `main()` yang akan dieksekusi saat program dijalankan.

5. `struct Node l1, l2, l3, l4, l5, l6, l7, l8, l9;` Mendeklarasikan sembilan variabel dari tipe data `struct Node` sebagai node-node dalam linked list.
6. `struct Node *link, *l3ptr;` Mendeklarasikan dua pointer ke tipe data `struct Node`.
7. `l1.link = NULL;` Menetapkan nilai `NULL` ke pointer `link` dari node `l1`.
8. `l1.alphabet = "F";` Menetapkan nilai string "F" ke atribut `alphabet` dari node `l1`.
9. Perintah-perintah serupa seperti di atas dilakukan untuk node-node `l2` hingga `l9`, masing-masing dengan nilai `alphabet` yang berbeda.
10. `l7.link = &l1;` Menetapkan pointer `link` dari node `l7` ke alamat node `l1`, sehingga membentuk sebuah linked list.
11. Perintah-perintah serupa seperti di atas dilakukan untuk menghubungkan node-node secara berurutan hingga terbentuk linked list yang lengkap.
12. `l3ptr = &l7;` Menetapkan alamat node `l7` ke pointer `l3ptr`.
13. Baris-baris berikutnya merupakan serangkaian perintah `printf` untuk mencetak nilai-nilai atribut `alphabet` dari beberapa node dalam linked list `l3`.

2.

Input :

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

```
int twoStacks(int maxSum, int a[], int n, int b[], int m) {
```

```
    int sum = 0, count = 0, temp = 0, i = 0, j = 0;
```

```
    while (i < n && sum + a[i] <= maxSum) {
```

```
        sum += a[i++];
```

```
    }
```

```
    count = i;
```

```
    while (j < m && i >= 0) {
```

```
        sum += b[j++];
```

```
        while (sum > maxSum && i > 0) {
```

```
            sum -= a[--i];
```

```
        }
```

```

        if (sum <= maxSum && i + j > count) {
            count = i + j;
        }
    }
    return count;
}

int main() {
    int g;
    scanf("%d", &g);
    while (g-- > 0) {
        int n, m, maxSum;
        scanf("%d%d%d", &n, &m, &maxSum);
        int a[n], b[m];
        for (int i = 0; i < n; i++) {
            scanf("%d", &a[i]);
        }
        for (int i = 0; i < m; i++) {
            scanf("%d", &b[i]);
        }
        printf("%d\n", twoStacks(maxSum, a, n, b, m));
    }
    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
---	---