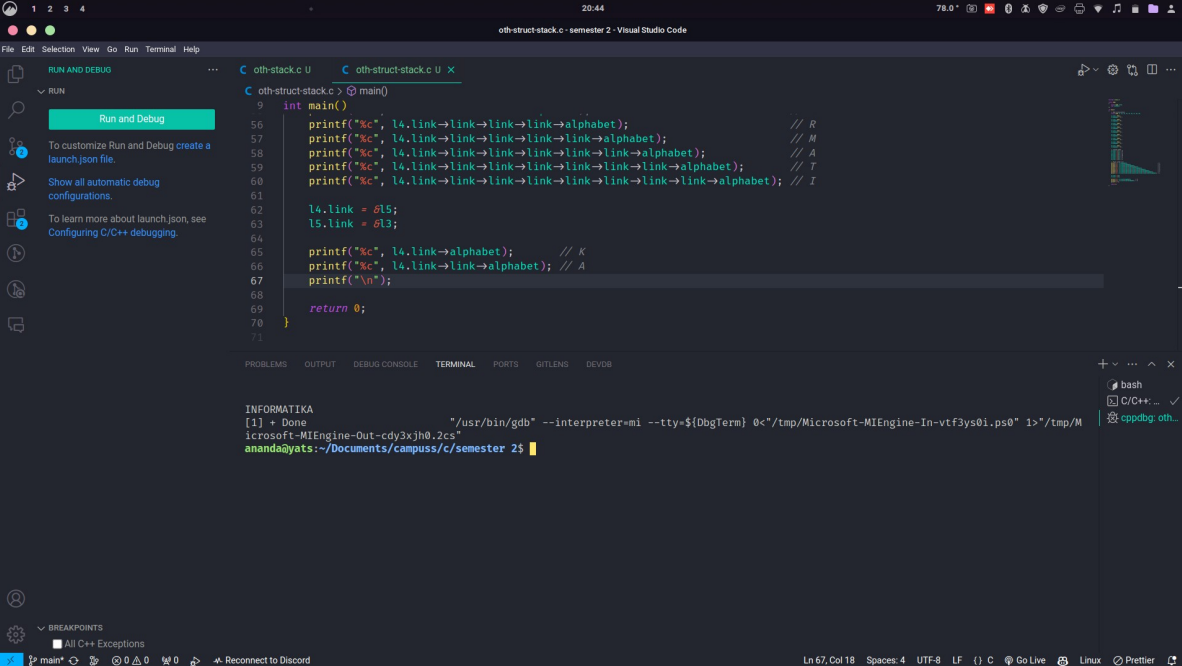


Nama : Muhammad Fikri Haikal Erhaz

Kelas : IF 03-02

NIM : 1203230104

Output Program



The screenshot shows the Visual Studio Code interface with a C++ program open in the editor. The program is a linked list implementation. The output window shows the execution results, which are the letters R, M, A, T, and I, each on a new line. The terminal window shows the command used to run the program: `g++ oth-stack.c -std=c++11 -g -o oth-stack`.

```
int main()
{
    printf("%c", l4.link->link->link->link->alphabet); // R
    printf("%c", l4.link->link->link->link->link->alphabet); // M
    printf("%c", l4.link->link->link->link->link->link->alphabet); // A
    printf("%c", l4.link->link->link->link->link->link->link->alphabet); // T
    printf("%c", l4.link->link->link->link->link->link->link->link->alphabet); // I

    l4.link = &l5;
    l5.link = &l3;

    printf("%c", l4.link->alphabet); // K
    printf("%c", l4.link->link->alphabet); // A
    printf("\n");

    return 0;
}
```

INFORMATIKA
[1] + Done
Microsoft-MIEngine-Out-cdy3xjh0.2cs
ananda@yats:~/Documents/campus/c/semester 2\$

1203230040_Ahanda Binta

Game of Two Stacks | Hack

https://www.hackerrank.com/challenges/game-of-two-stacks/problem

```
235 int parse_int(char *str)
236 {
237     char *endptr;
238     int value = strtol(str, &endptr, 10);
239
240     if (endptr == str || *endptr != '\0')
241     {
242         exit(EXIT_FAILURE);
243     }
244     return value;
245 }
247
```

Line: 247 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

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)

Download

```
1 4
```

Expected Output

Download

```
1 4
```

1203230040_Ahanda Binta

Game of Two Stacks | Hack

https://www.hackerrank.com/challenges/game-of-two-stacks/problem

```
237 char *endptr;
238 int value = strtol(str, &endptr, 10);
239
240 if (endptr == str || *endptr != '\0')
241 {
242     exit(EXIT_FAILURE);
243 }
244 return value;
245 }
247
```

Line: 247 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Test case 6

Compiler Message

Success

Input (stdin)

Download

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

Expected Output

Download

```
1 4
```

Penjelasan Kode

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

struct node *link: Anggota ini adalah pointer yang menunjukkan ke node berikutnya dalam struktur data yang terkait. Ini adalah dasar dari pembentukan struktur data yang disebut linked list. Pointer ini menghubungkan node-node bersama dalam urutan tertentu.

char alphabet: Anggota ini adalah karakter yang menyimpan nilai huruf. Ini mungkin digunakan untuk menyimpan informasi tambahan dalam setiap node, misalnya, jika struktur data ini digunakan untuk membuat linked list yang menyimpan karakter

```

// Node initialization
struct node l1, l2, l3, l4, l5, l6, l7, l8, l9;

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';

// Linking nodes
l4.link = &l7; // N
l7.link = &l1; // F
l1.link = &l8; // O
l8.link = &l9; // R
l9.link = &l2; // M
l2.link = &l3; // A
l3.link = &l6; // T
l6.link = &l4; // I

```

Menginisialisasi variabel bertipe data “struct node” l1-l9 untuk assign setiap huruf. Lalu link/menghubungkan setiap nodes agar menjadi kata yang diinginkan.

```

// Print linked list
printf("%c", l4.alphabet);           // I
printf("%c", l4.link→alphabet);     // N
printf("%c", l4.link→link→alphabet); // F
printf("%c", l4.link→link→link→alphabet); // O
printf("%c", l4.link→link→link→link→alphabet); // R
printf("%c", l4.link→link→link→link→link→alphabet); // M
printf("%c", l4.link→link→link→link→link→link→alphabet); // A
printf("%c", l4.link→link→link→link→link→link→link→alphabet); // T
printf("%c", l4.link→link→link→link→link→link→link→link→alphabet); // I

l4.link = &l5;
l5.link = &l3;

printf("%c", l4.link→alphabet);      // K
printf("%c", l4.link→link→alphabet); // A
printf("\n");

```

Output nodes yang telah dilink/dihubungkan menjadi satu kesatuan kata.

```
19 //
20 * Complete the 'twoStacks' function below.
21 *
22 * The function is expected to return an INTEGER.
23 * The function accepts following parameters:
24 * 1. INTEGER maxSum
25 * 2. INTEGER_ARRAY a
26 * 3. INTEGER_ARRAY b
27 */
28
29 int twoStacks(int maxSum, int a_count, int *a, int b_count, int *b)
30 {
31     int i = 0, j = 0, sum = 0, count = 0;
32     while (i < a_count && sum + a[i] <= maxSum)
33     {
34         sum += a[i];
35         i++;
36     }
37     count = i;
38     while (j < b_count && i > 0)
39     {
40         sum += b[j];
41         j++;
42         while (sum > maxSum && i > 0)
43         {
44             i--;
45             sum -= a[i];
46         }
47         if (sum <= maxSum && i + j > count)
48         {
49             count = i + j;
50         }
51     }
52     return count;
53 }
54
55 int main()
56 {
57     FILE *fptr = fopen(getenv("OUTPUT_PATH"), "w");
```

1. Iterasi melalui stack pertama (a) sampai total elemen (sum) tidak melebihi maxSum. Di setiap langkah, tambahkan elemen saat ini dari stack pertama ke sum dan tambahkan 1 ke penghitung i.
2. Inisialisasi count dengan nilai i saat ini, yang menunjukkan berapa banyak elemen yang telah diambil dari stack pertama.
3. Iterasi melalui stack kedua (b). Di setiap langkah, tambahkan elemen saat ini dari stack kedua ke sum dan tambahkan 1 ke penghitung j.
4. Jika sum melebihi maxSum dan masih ada elemen di stack pertama, hapus elemen terakhir dari stack pertama untuk menjaga sum tidak melebihi maxSum.
5. Setelah selesai iterasi, perbarui count jika sum masih dalam batas maxSum dan jumlah elemen dari kedua stack lebih besar dari count saat ini.
6. Akhirnya, kembalikan nilai count sebagai jumlah maksimum elemen yang dapat diambil dari kedua stack tanpa melebihi maxSum.