

Nama: Muhammad Muhyi Akbar

Nim: 1203230090

IF 02

1. Source code

```
2. #include <stdio.h>
3.
4. struct node // Tipe data baru
5. {
6.     struct node *link;
7.     char alphabet;
8. };
9.
10. int main()
11. {
12.     // Inisialisasi Node
13.     struct node l1, l2, l3, l4, l5, l6, l7, l8, l9;
14.
15.     l1.link = NULL;
16.     l1.alphabet = 'F';
17.
18.     l2.link = NULL;
19.     l2.alphabet = 'M';
20.
21.     l3.link = NULL;
22.     l3.alphabet = 'A';
23.
24.     l4.link = NULL;
25.     l4.alphabet = 'I';
26.
27.     l5.link = NULL;
28.     l5.alphabet = 'K';
29.
30.     l6.link = NULL;
31.     l6.alphabet = 'T';
32.
33.     l7.link = NULL;
34.     l7.alphabet = 'N';
35.
36.     l8.link = NULL;
37.     l8.alphabet = 'O';
38.
39.     l9.link = NULL;
40.     l9.alphabet = 'R';
41.
```

```

42. // Linking nodes
43. 14.link = &l7; // I > N
44. 17.link = &l1; // N > F
45. 11.link = &l8; // F > O
46. 18.link = &l9; // O > R
47. 19.link = &l2; // R > M
48. 12.link = &l3; // M > A
49. 13.link = &l6; // A > T
50. 16.link = &l4; // T > I
51.
52. // Print linked list
53. printf("%c",
    14.alphabet); // I
54. printf("%c", 14.link-
    >alphabet); // N
55. printf("%c", 14.link->link-
    >alphabet); // F
56. printf("%c", 14.link->link->link-
    >alphabet); // O
57. printf("%c", 14.link->link->link->link-
    >alphabet); // R
58. printf("%c", 14.link->link->link->link->link-
    >alphabet); // M
59. printf("%c", 14.link->link->link->link->link->link-
    >alphabet); // A
60. printf("%c", 14.link->link->link->link->link->link->link-
    >alphabet); // T
61. printf("%c", 14.link->link->link->link->link->link->link->link-
    >alphabet); // I
62.
63. // Dipisah agar tidak mengubah nilai yang telah dibuat terlebih
    dahulu
64. 14.link = &l5; // I > K
65. 15.link = &l3; // K > A
66.
67. printf("%c", 14.link-
    >alphabet); // K
68. printf("%c", 14.link->link-
    >alphabet); // A
69. printf("\n");
70.
71. return 0;
72.}

```

The screenshot shows a web browser with the URL <https://www.programiz.com/c-programming/online-compiler/>. The page features a banner for 'Silaturahmi #RameDanSeru' and a 'Programiz PRO' button. The main content area displays a C program in a file named 'main.c'. The code defines a linked list structure and a function to traverse it, printing the characters 'A', 'T', and 'I'. The output on the right shows 'INFORMATIKA' and a message '=== Code Execution Successful ==='.

```

main.c
58 printf("%c", 14.link->link->link->link->link
   ->alphabet); // A
59 printf("%c", 14.link->link->link->link->link->link->link
   ->alphabet); // T
60 printf("%c", 14.link->link->link->link->link->link->link->link
   ->link->alphabet); // I
61
62 // Dipisah agar tidak mengubah nilai yang telah dibuat
   terlebih dahulu
63 14.link = &l5; // I > K
64 15.link = &l3; // K > A
65
66 printf("%c", 14.link->alphabet);
   // K
67 printf("%c", 14.link->link->alphabet);
   // A
68 printf("\n");
69
70 return 0;
71 }

```

Output

```

/tmp/tmp5NNX2fCo.o
INFORMATIKA

=== Code Execution Successful ===

```

Kemudian, pointer link antar node ditetapkan untuk membentuk linked list. Setiap node terhubung ke node berikutnya sesuai dengan urutan penomoran yang diberikan.

Terakhir, dilakukan pencetakan karakter-karakter dalam linked list lagi setelah perubahan.

2. Source code

```
73.#include <assert.h>
74.#include <ctype.h>
75.#include <limits.h>
76.#include <math.h>
77.#include <stdbool.h>
78.#include <stddef.h>
79.#include <stdint.h>
80.#include <stdio.h>
81.#include <stdlib.h>
82.#include <string.h>
83.
84.char *getline();
85.char *ltrim(char *);
86.char *rtrim(char *);
87.char **split_string(char *);
88.
89.int parse_int(char *);
90.
91.int twoStacks(int maxSum, int a_count, int *a, int b_count, int *b)
92.{
93.    int i = 0, j = 0, sum = 0, count = 0;
94.    while (i < a_count && sum + a[i] <= maxSum)
95.    {
96.        sum += a[i];
97.        i++;
98.    }
99.    count = i;
100.    while (j < b_count && i >= 0)
101.    {
102.        sum += b[j];
103.        j++;
104.        while (sum > maxSum && i > 0)
105.        {
106.            i--;
107.            sum -= a[i];
108.        }
109.        if (sum <= maxSum && i + j > count)
110.        {
111.            count = i + j;
112.        }
113.    }
114.    return count;
115.}
116.
117.int main()
```

```
118.     {
119.         FILE *fptr = fopen(getenv("OUTPUT_PATH"), "w");
120.
121.         int g = parse_int(ltrim(rtrim(readline())));
122.
123.         for (int g_itr = 0; g_itr < g; g_itr++)
124.         {
125.             char **first_multiple_input =
split_string(rtrim(readline()));
126.
127.             int n = parse_int(*(first_multiple_input + 0));
128.
129.             int m = parse_int(*(first_multiple_input + 1));
130.
131.             int maxSum = parse_int(*(first_multiple_input + 2));
132.
133.             char **a_temp = split_string(rtrim(readline()));
134.
135.             int *a = malloc(n * sizeof(int));
136.
137.             for (int i = 0; i < n; i++)
138.             {
139.                 int a_item = parse_int(*(a_temp + i));
140.
141.                 *(a + i) = a_item;
142.             }
143.
144.             char **b_temp = split_string(rtrim(readline()));
145.
146.             int *b = malloc(m * sizeof(int));
147.
148.             for (int i = 0; i < m; i++)
149.             {
150.                 int b_item = parse_int(*(b_temp + i));
151.
152.                 *(b + i) = b_item;
153.             }
154.
155.             int result = twoStacks(maxSum, n, a, m, b);
156.
157.             fprintf(fptr, "%d\n", result);
158.         }
159.
160.         fclose(fptr);
161.
162.         return 0;
163.     }
164.
```

```

165. char *readline()
166. {
167.     size_t alloc_length = 1024;
168.     size_t data_length = 0;
169.
170.     char *data = malloc(alloc_length);
171.
172.     while (true)
173.     {
174.         char *cursor = data + data_length;
175.         char *line = fgets(cursor, alloc_length - data_length,
stdin);
176.
177.         if (!line)
178.         {
179.             break;
180.         }
181.
182.         data_length += strlen(cursor);
183.
184.         if (data_length < alloc_length - 1 || data[data_length -
1] == '\n')
185.         {
186.             break;
187.         }
188.
189.         alloc_length <= 1;
190.
191.         data = realloc(data, alloc_length);
192.
193.         if (!data)
194.         {
195.             data = '\0';
196.
197.             break;
198.         }
199.     }
200.
201.     if (data[data_length - 1] == '\n')
202.     {
203.         data[data_length - 1] = '\0';
204.
205.         data = realloc(data, data_length);
206.
207.         if (!data)
208.         {
209.             data = '\0';
210.         }

```

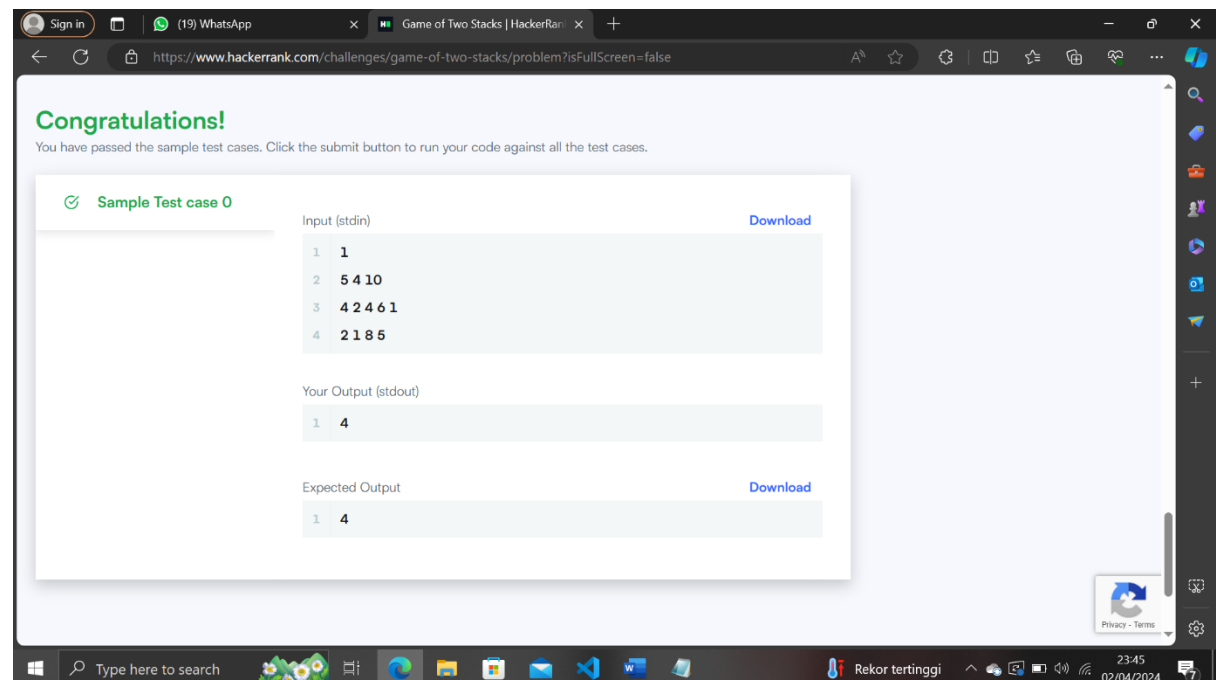
```
211.     }
212.     else
213.     {
214.         data = realloc(data, data_length + 1);
215.
216.         if (!data)
217.         {
218.             data = '\0';
219.         }
220.         else
221.         {
222.             data[data_length] = '\0';
223.         }
224.     }
225.
226.     return data;
227. }
228.
229. char *ltrim(char *str)
230. {
231.     if (!str)
232.     {
233.         return '\0';
234.     }
235.
236.     if (!*str)
237.     {
238.         return str;
239.     }
240.
241.     while (*str != '\0' && isspace(*str))
242.     {
243.         str++;
244.     }
245.
246.     return str;
247. }
248.
249. char *rtrim(char *str)
250. {
251.     if (!str)
252.     {
253.         return '\0';
254.     }
255.
256.     if (!*str)
257.     {
258.         return str;
```

```
259.     }
260.
261.     char *end = str + strlen(str) - 1;
262.
263.     while (end >= str && isspace(*end))
264.     {
265.         end--;
266.     }
267.
268.     *(end + 1) = '\0';
269.
270.     return str;
271. }
272.
273. char **split_string(char *str)
274. {
275.     char **splits = NULL;
276.     char *token = strtok(str, " ");
277.
278.     int spaces = 0;
279.
280.     while (token)
281.     {
282.         splits = realloc(splits, sizeof(char *) * ++spaces);
283.
284.         if (!splits)
285.         {
286.             return splits;
287.         }
288.
289.         splits[spaces - 1] = token;
290.
291.         token = strtok(NULL, " ");
292.     }
293.
294.     return splits;
295. }
296.
297. int parse_int(char *str)
298. {
299.     char *endptr;
300.     int value = strtol(str, &endptr, 10);
301.
302.     if (endptr == str || *endptr != '\0')
303.     {
304.         exit(EXIT_FAILURE);
305.     }
306.
```



```
307.         return value;
308.     }
```

Output:



Penjelasan:

`twoStacks(int maxSum, int a_count, int *a, int b_count, int *b)`: Fungsi ini mengambil lima parameter, yaitu nilai maksimum (`maxSum`), jumlah elemen dalam tumpukan pertama (`a_count`), array yang mewakili tumpukan pertama (`a`), jumlah elemen dalam tumpukan kedua (`b_count`), dan array yang mewakili tumpukan kedua (`b`). Fungsi ini mengembalikan jumlah maksimum elemen yang dapat diambil dari kedua tumpukan tanpa melebihi `maxSum`.

`main()`: Fungsi utama dari program ini. Pertama-tama, membuka file keluaran dan membaca jumlah kasus uji (`g`). Kemudian, untuk setiap kasus uji, membaca input berupa jumlah elemen dalam tumpukan pertama (`n`), jumlah elemen dalam tumpukan kedua (`m`), dan nilai `maxSum`. Selanjutnya, membaca elemen-elemen tumpukan pertama dan kedua, memanggil fungsi `twoStacks`, dan menulis hasilnya ke file keluaran. Setelah selesai, menutup file keluaran.

`getline()`: Fungsi ini membaca satu baris dari input standar (`stdin`) dan mengembalikan string yang berisi baris tersebut. Fungsi ini secara dinamis mengalokasikan memori seiring bertambahnya panjang baris.

`ltrim(char *str)`: Fungsi ini menghapus spasi (`whitespace`) dari awal string (`str`) dan mengembalikan pointer ke string tersebut.

`rtrim(char *str)`: Fungsi ini menghapus spasi (`whitespace`) dari akhir string (`str`) dan mengembalikan pointer ke string tersebut.

`split_string(char *str)`: Fungsi ini memecah string (`str`) menjadi potongan-potongan berdasarkan spasi, kemudian mengembalikan array pointer ke potongan-potongan tersebut.

`parse_int(char *str)`: Fungsi ini mengonversi string (`str`) menjadi bilangan bulat dan mengembalikan nilainya. Jika konversi gagal, program akan keluar dengan status kesalahan.