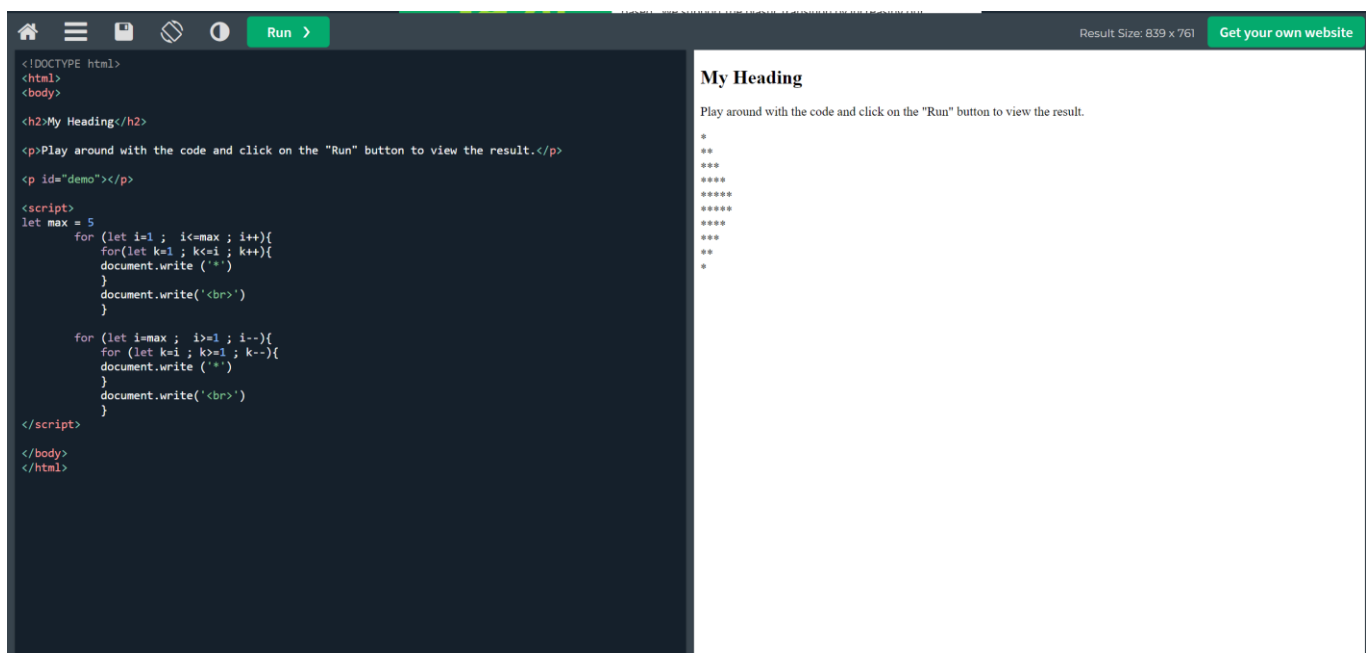


Nama : Leody Zelvon Herliansa

NRP : 3122500010

Kelas : D3 IT A

## Program Gambar Segitiga



The screenshot shows a web browser interface. On the left, there is a code editor with the following HTML and JavaScript code:

```
<!DOCTYPE html>
<html>
<body>

<h2>My Heading</h2>

<p>Play around with the code and click on the "Run" button to view the result.</p>
<p id="demo"></p>
<script>
let max = 5
  for (let i=1 ; i<=max ; i++){
    for(let k=1 ; k<=i ; k++){
      document.write ('*')
    }
    document.write('<br>')
  }

  for (let i=max ; i>=1 ; i--){
    for (let k=i ; k>=1 ; k--){
      document.write ('*')
    }
    document.write('<br>')
  }
</script>
</body>
</html>
```

On the right, the browser displays the rendered page. It features a heading "My Heading" and a paragraph: "Play around with the code and click on the "Run" button to view the result." Below this, a triangle of asterisks is displayed, with 5 rows: the first row has 1 asterisk, the second has 2, the third has 3, the fourth has 4, and the fifth has 5. The browser's top bar shows "Result Size: 839 x 761" and a "Get your own website" button.

## Penjelasan

Pada tag body ada property onload memanggil fungsi segitiga dengan memberikan parameter 5 artinya ketika halaman diload akan memanggil fungsi segitiga

Fungsi segitiga menerima parameter height. kemudian dalam fungsi segitiga variable hasil dengan jenis let. mengapa menggunakan let? Karena let merupakan jenis variable dalam javascript yang dapat diubah nilainya dan menganut sistem scope yang artinya variable ini hanya bisa diakses dalam sebuah scope saja

Kemudian ke blok perulangan for. dalam for ini akan melakukan perulangan selama variable i lebih kecil dari height dan akan diincrement

Masuk ke dalam for nya lagi akan mencetak karakter # sejumlah nilai variable i

Ke for yang dibawah logic sama seperti for yang diatas namun yang membedakan adalah variable I akan di decrement

## Hasil compile Assembly(Bahasa Mesin) dari bahasa C++ dengan compiler x86-64 gcc 12.2

Kode Bahasa C++

```
main.cpp x
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      cout << "Hello world!" << endl;
8      return 0;
9  }
10
```

Kode Assembly(Bahasa Mesin)

```
.LC0:
    .string "hello world"
main:
    push    rbp
    mov     rbp, rsp
    mov     edi, OFFSET FLAT:.LC0
    mov     eax, 0
    call   printf
    mov     eax, 0
    pop     rbp
    ret
__static_initialization_and_destruction_0(int, int):
    push    rbp
    mov     rbp, rsp
    sub     rsp, 16
    mov     DWORD PTR [rbp-4], edi
    mov     DWORD PTR [rbp-8], esi
    cmp     DWORD PTR [rbp-4], 1
    jne     .L5
    cmp     DWORD PTR [rbp-8], 65535
    jne     .L5
    mov     edi, OFFSET FLAT:_ZStL8__ioinit
    call   std::ios_base::Init::Init() [complete object constructor]
    mov     edx, OFFSET FLAT:__dso_handle
    mov     esi, OFFSET FLAT:_ZStL8__ioinit
    mov     edi, OFFSET FLAT:_ZNSt8ios_base4InitD1Ev
```

```

        call    __cxa_atexit
.L5:
        nop
        leave
        ret
_GLOBAL__sub_I_main:
        push   rbp
        mov    rbp, rsp
        mov    esi, 65535
        mov    edi, 1
        call   __static_initialization_and_destruction_0(int, int)
        pop   rbp
        ret

```

## Penjelasan

- Bagian *LCO* adalah section untuk mengalamatkan konstanta string
  - Main: adalah section utama dalam sebuah program
  - Bagian *\_\_static\_initialization\_and\_destruction\_0(int, int)*: adalah method construction statis yang digunakan apabila membutuhkan method construction statis (jika dalam OOP construction adalah method yang dijalankan ketika sebuah objek diinstansiasi objek)
  - .L5 adalah jump table digunakan untuk mengirim program control ke bagian lain
  - *\_GLOBAL\_\_sub\_I\_main*: Fungsi untuk scoping sederhana disekitarnya
1. CMP: digunakan untuk membandingkan 2 nilai/value
  2. Push : mendorong nilai (tidak harus disimpan dalam register) berarti menulisnya ke stack
  3. Call : memanggil sebuah fungsi
  4. Pop : mengembalikan apa pun yang ada di atas stack ke dalam register
  5. MOV digunakan untuk register ke register pengalamatan
  6. Sub : mendapatkan shift, operasi bitwise dan perkalian/pembagian dengan mudah.
  7. JNE : (jump if not equal) mengeksekusi lompatan jika flag sama dengan nol.
  8. NOP : pernyataan atau perintah protokol komputer yang tidak melakukan apa pun
  9. Ret : mentransfer kontrol ke alamat pengirim yang terletak di stack