

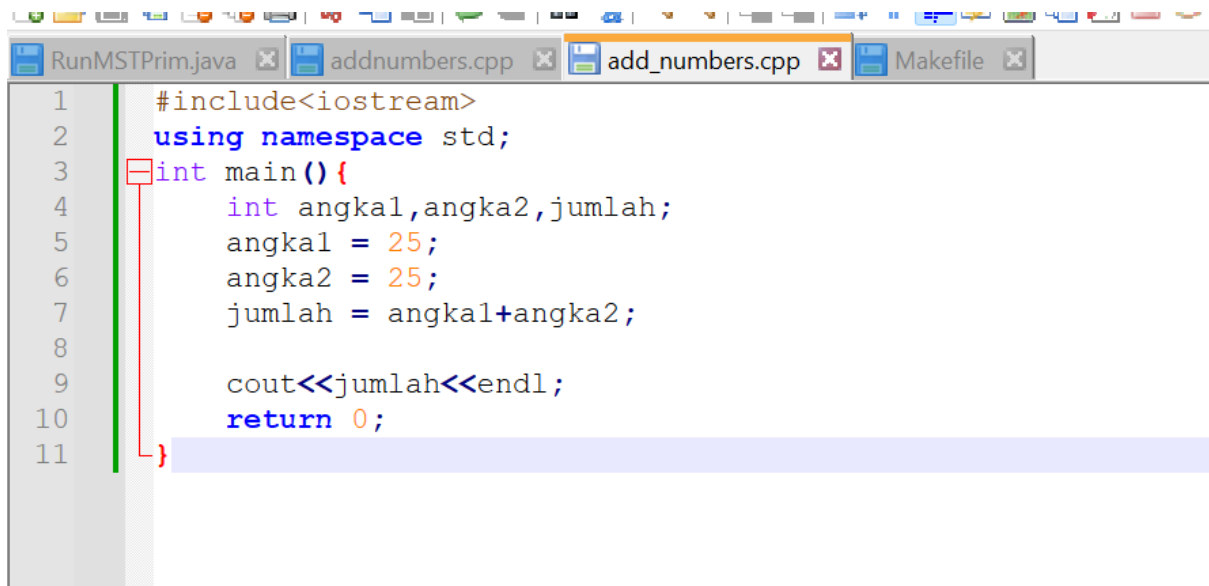
Nama : Bagus Cipta Pratama

Nim : 23/516539/PA/22097

Assignment 1

Link github : <https://github.com/mobssspro/Bagus-Cipta-Pratama-SKJ-Lab.git>

nomor 1 :



```
1  #include<iostream>
2  using namespace std;
3  int main() {
4      int angka1,angka2,jumlah;
5      angka1 = 25;
6      angka2 = 25;
7      jumlah = angka1+angka2;
8
9      cout<<jumlah<<endl;
10     return 0;
11 }
```

The image shows a screenshot of a C++ code editor with four tabs: 'RunMSTPrim.java', 'addnumbers.cpp', 'add_numbers.cpp', and 'Makefile'. The 'add_numbers.cpp' tab is active, displaying a C++ program. The code defines a main function that declares three integer variables: 'angka1', 'angka2', and 'jumlah'. It assigns the value 25 to both 'angka1' and 'angka2', then calculates their sum and stores it in 'jumlah'. Finally, it prints the value of 'jumlah' and returns 0. The code is line-numbered from 1 to 11. A red bracket on the left side of the editor highlights the entire main function block, spanning from line 3 to line 11.

```
C:\Users\ACER\Assignment1\nomor1>start notepad++ add_numbers.cpp

C:\Users\ACER\Assignment1\nomor1>g++ -o add_numbers add_numbers.cpp

C:\Users\ACER\Assignment1\nomor1>add_numbers
50
```

```
C:\Users\ACER\Assignment1\nomor1>objdump -d add_numbers
objdump: 'add_numbers': No such file
```

```
C:\Users\ACER\Assignment1\nomor1>objdump -d add_numbers.exe
```

```
C:\Users\ACER\Assignment1\nomor1>objdump -d add_numbers
objdump: 'add_numbers': No such file
```

```
C:\Users\ACER\Assignment1\nomor1>objdump -d add_numbers.exe
```

```
add_numbers.exe:      file format pei-i386
```

Disassembly of section .text:

```
00401000 <.text>:
 401000:      83 ec 1c          sub     $0x1c,%esp
 401003:      8b 44 24 20       mov     0x20(%esp),%eax
 401007:      8b 00             mov     (%eax),%eax
 401009:      8b 00             mov     (%eax),%eax
 40100b:      3d 91 00 00 c0     cmp     $0xc0000091,%eax
 401010:      77 4e             ja      401060 <.text+0x60>
 401012:      3d 8d 00 00 c0     cmp     $0xc000008d,%eax
 401017:      73 60             jae     401079 <.text+0x79>
 401019:      3d 05 00 00 c0     cmp     $0xc0000005,%eax
 40101e:      0f 85 cc 00 00 00   jne     4010f0 <.text+0xf0>
 401024:      c7 44 24 04 00 00 00 movl    $0x0,0x4(%esp)
 40102b:      00
 40102c:      c7 04 24 0b 00 00 00 movl    $0xb,(%esp)
 401033:      e8 e0 2a 00 00     call    403b18 <_signal>
 401038:      83 f8 01          cmp     $0x1,%eax
 40103b:      0f 84 48 01 00 00   je      401189 <.text+0x189>
 401041:      85 c0             test    %eax,%eax
 401043:      0f 85 e7 00 00 00   jne     401130 <.text+0x130>
 401049:      8d b4 26 00 00 00 00 lea     0x0(%esi,%eiz,1),%esi
 401050:      31 c0             xor     %eax,%eax
 401052:      83 c4 1c          add     $0x1c,%esp
 401055:      c2 04 00          ret     $0x4
 401058:      90              nop
```

```

00401460 <_main>:
  401460:      8d 4c 24 04      lea     0x4(%esp),%ecx
  401464:      83 e4 f0         and     $0xffffffff0,%esp
  401467:      ff 71 fc         pushl   -0x4(%ecx)
  40146a:      55                push     %ebp
  40146b:      89 e5             mov     %esp,%ebp
  40146d:      51                push     %ecx
  40146e:      83 ec 24         sub     $0x24,%esp
  401471:      e8 0a 06 00 00    call    401a80 <___main>
  401476:      c7 45 f4 19 00 00 00 movl    $0x19,-0xc(%ebp)
  40147d:      c7 45 f0 19 00 00 00 movl    $0x19,-0x10(%ebp)
  401484:      8b 55 f4         mov     -0xc(%ebp),%edx
  401487:      8b 45 f0         mov     -0x10(%ebp),%eax
  40148a:      01 d0            add     %edx,%eax
  40148c:      89 45 ec         mov     %eax,-0x14(%ebp)
  40148f:      8b 45 ec         mov     -0x14(%ebp),%eax
  401492:      89 04 24         mov     %eax,(%esp)
  401495:      b9                .byte 0xb9

```

add_numbers.exe: file format pei-i386

Disassembly of section .text:

```

00401000 <.text>:
  401000:      83 ec 1c         sub     $0x1c,%esp
  401003:      8b 44 24 20      mov     0x20(%esp),%eax
  401007:      8b 00            mov     (%eax),%eax
  401009:      8b 00            mov     (%eax),%eax
  40100b:      3d 91 00 00 c0    cmp     $0xc0000091,%eax
  401010:      77 4e            ja      401060 <.text+0x60>
  401012:      3d 8d 00 00 c0    cmp     $0xc000008d,%eax
  401017:      73 60            jae     401079 <.text+0x79>
  401019:      3d 05 00 00 c0    cmp     $0xc0000005,%eax
  40101e:      0f 85 cc 00 00 00 jne     4010f0 <.text+0xf0>
  401024:      c7 44 24 04 00 00 00 movl    $0x0,0x4(%esp)
  40102b:      00
  40102c:      c7 04 24 0b 00 00 00 movl    $0xb,(%esp)
  401033:      e8 e0 2a 00 00    call    403b18 <_signal>
  401038:      83 f8 01         cmp     $0x1,%eax
  40103b:      0f 84 48 01 00 00 je      401189 <.text+0x189>
  401041:      85 c0            test    %eax,%eax

```

```

C:\Users\ACER>cd assignment1

C:\Users\ACER\Assignment1>cd nomor1

C:\Users\ACER\Assignment1\nomor1>make run
g++ -o add_numbers add_numbers.cpp
./add_numbers
50

C:\Users\ACER\Assignment1\nomor1>make dump
g++ -o add_numbers add_numbers.cpp
objdump -d add_numbers.exe > add_numbers.asm

C:\Users\ACER\Assignment1\nomor1>add_numbers.asm

C:\Users\ACER\Assignment1\nomor1>make all
g++ -o add_numbers add_numbers.cpp

C:\Users\ACER\Assignment1\nomor1>make run
g++ -o add_numbers add_numbers.cpp
./add_numbers
50

C:\Users\ACER\Assignment1\nomor1>|

```

nomor 2 :

1. Penjelasan kode assembly :

Pada mulanya di section .data kita menyimpan berbagai variable yang ada . disini num1 dw 5 menyatakan num1 disimpan dengan 5 , begitu juga num 2 yang disimpan dengan nilai 10 sedangkan disini variable result diinisialisasi dengan nilai 0 .

section.text adalah segmen bagian kode mesin (machine code)

global _start berfungsi untuk membuat label '_start' sehingga bisa diakses dari luar , yang berarti ini adalah titik masuk (entry point) dari program

dalam _start ada beberapa instruksi `mov ax, [num1]` memindahkan nilai dari num1 ke register ax. `imul ax, [num2]` mengalikan nilai ax dengan nilai dari num2, hasilnya disimpan kembali di ax. `mov [result], ax` memindahkan hasil perkalian yang ada di ax ke variabel result.

Terakhir pada ; exit the program (bagian akhir dari kode yang keluar) , `mov eax, 1` mengatur register eax dengan nilai 1, yang berarti sistem call exit akan dipanggil. `xor ebx, ebx` mengatur register ebx menjadi 0, yang merupakan kode status keluarnya. `int 0x80` memanggil interrupt untuk menjalankan sistem call.

2. Perkalian.cpp file(hasil dari kode assembly) :

```
#include <iostream>
using namespace std;
int main() {
    int num1 = 5;
    int num2 = 10;
    int result = 0;

    result = num1 * num2;
    cout << "Hasil : " << result << endl;
    return 0;
}
```

3. Write 'Makefile' :

```
C:\Users\ACER\Assignment1\nomor2>make all
g++ -o perkalian perkalian.cpp

C:\Users\ACER\Assignment1\nomor2>make clean
rm -f perkalian perkalian.asm

C:\Users\ACER\Assignment1\nomor2>make dump
g++ -o perkalian perkalian.cpp
objdump -d perkalian > perkalian.asm
objdump: 'perkalian': No such file
make: *** [dump] Error 1

C:\Users\ACER\Assignment1\nomor2>make run
g++ -o perkalian perkalian.cpp
./perkalian
Hasil : 50
```