PRAKTIKUM SISTEM OPERASI

MODUL 1(Tugas 1)

Nama : NICKY JULYATRIKA SARI

NIM : L200200101

PROGRAM STUDI

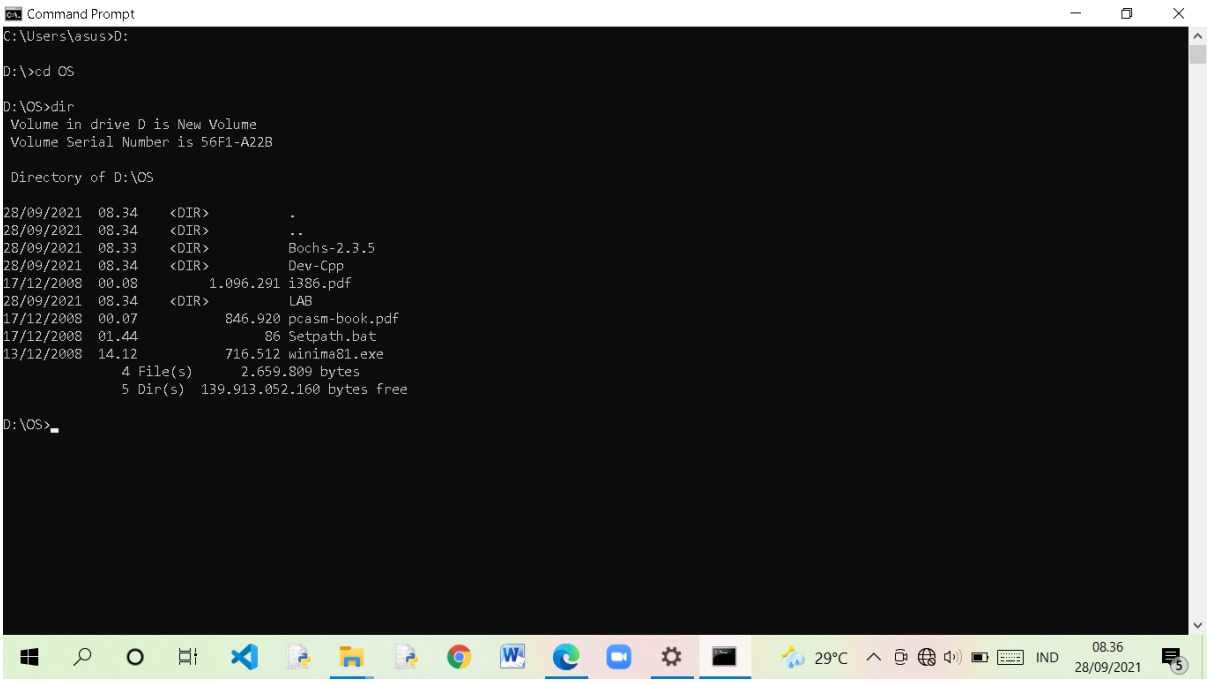
INFORMATIKA

FAKULTAS KOMUNIKASI DAN INFORMATIKA

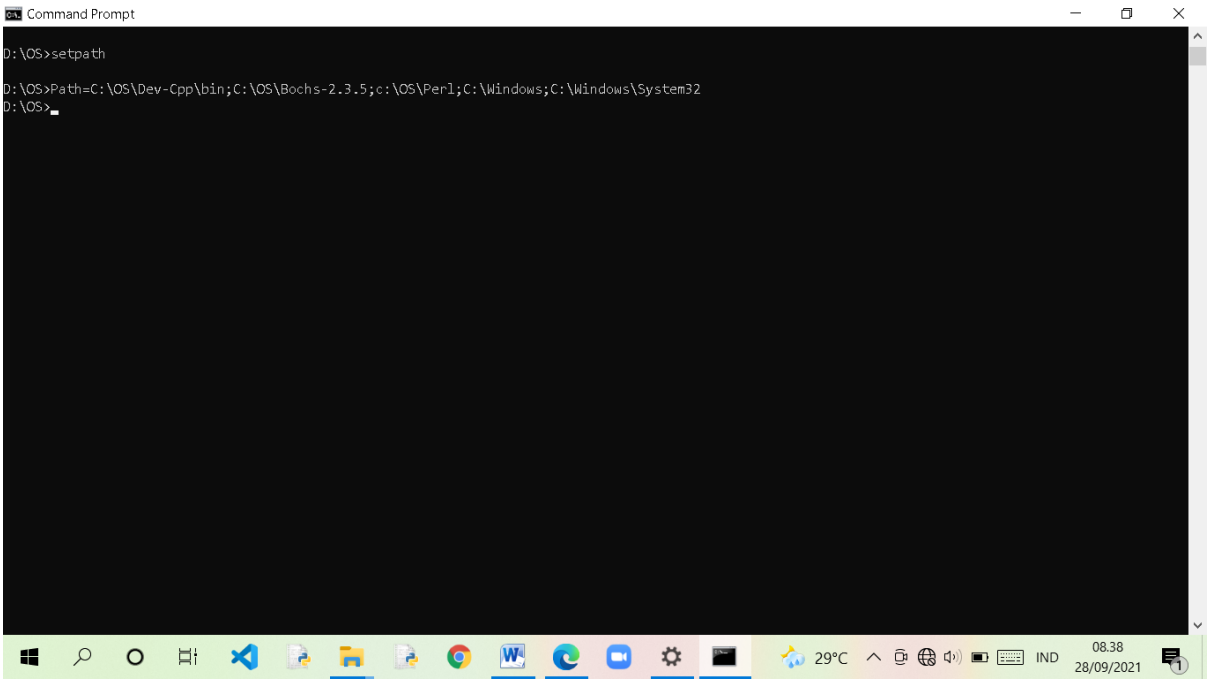
UNIVERSITAS MUHAMMADIYAH SURAKARTA

TAHUN 2021/2022

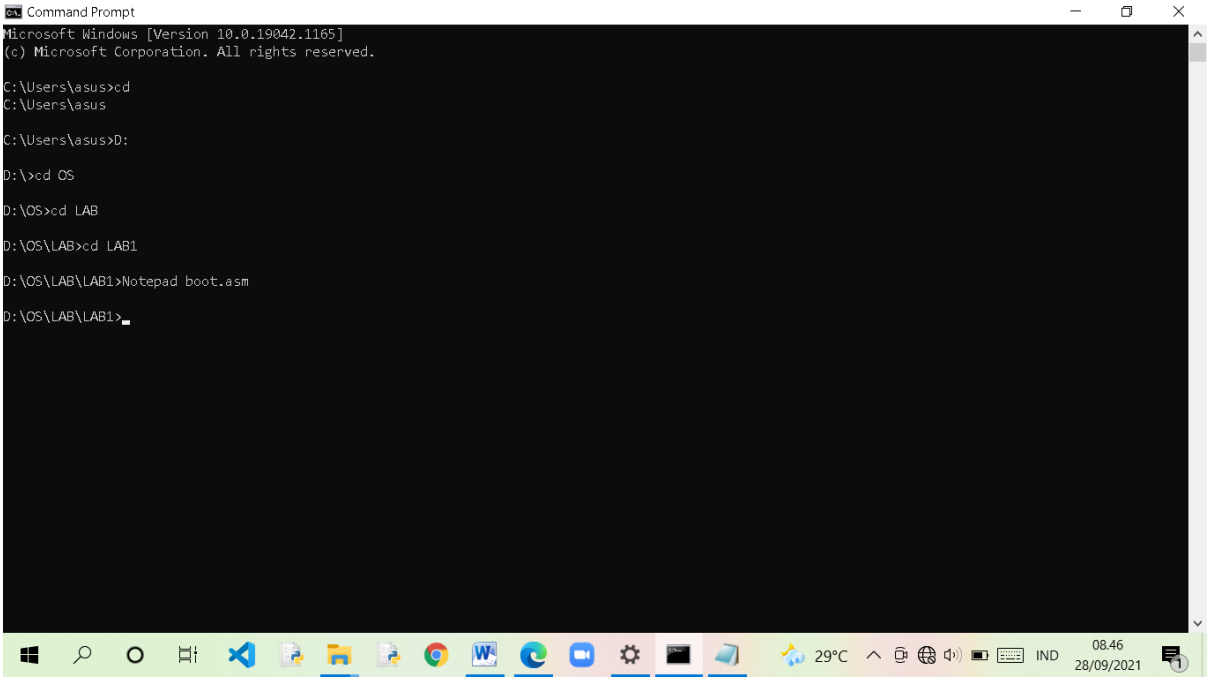
1. Menuju ke direktori kerja
   a) jalankan program command promt atau cmd.
   b) Masuk ke direktori kerja ’C:\OS’, dengan perintah ’cd os’ .
   c) Masukan perintah dir, untuk melihat isi direktori di dalam folder tersebut. Akan muncul seperti di tampilkan pada gambar dibawah

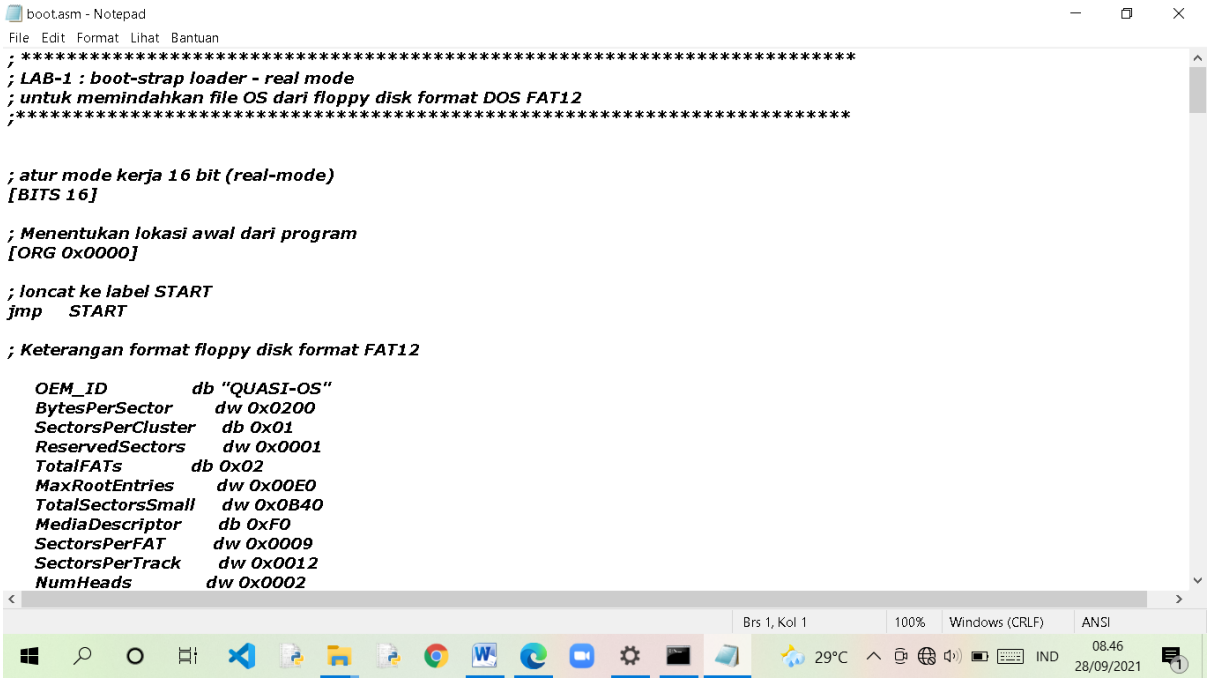


   d) Jalankan file setpath, untuk menjalankannya ketik ‘setpath’ tekan

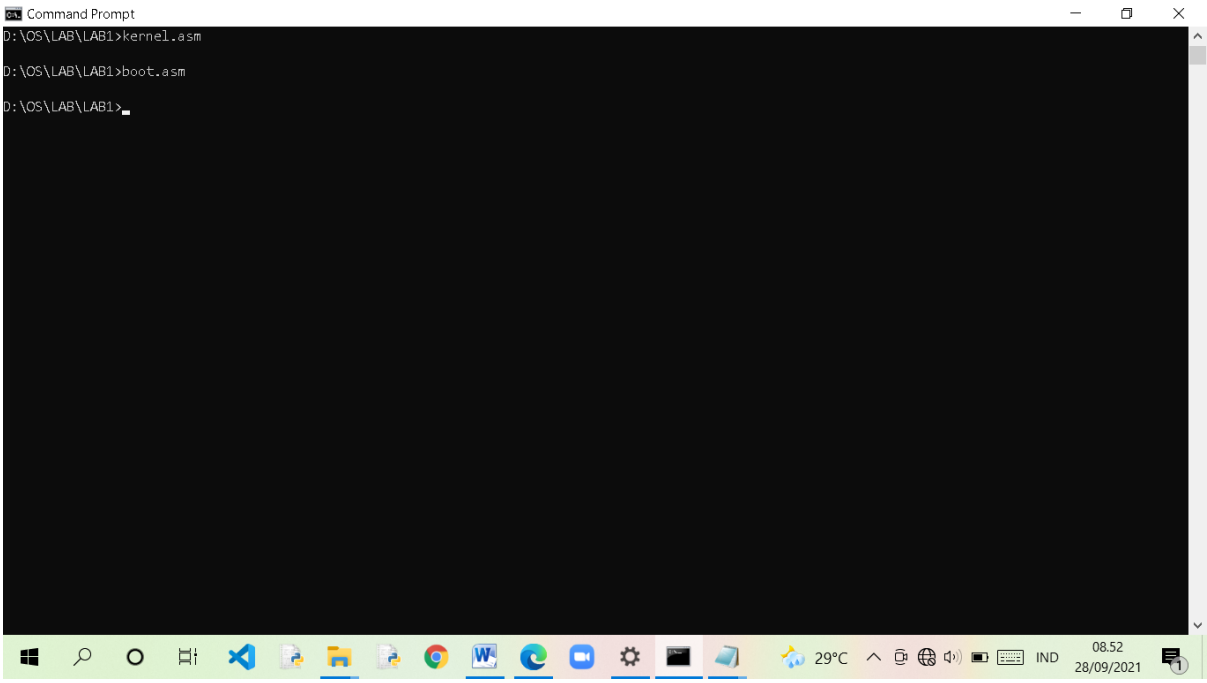


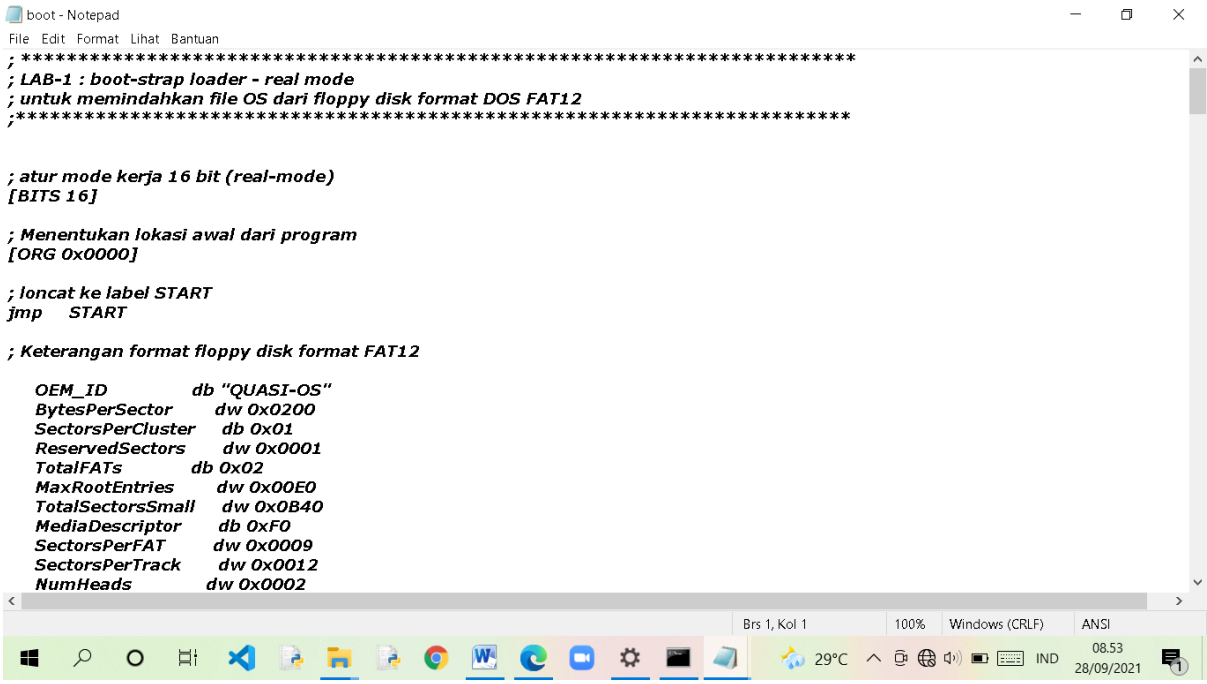
1. Melihat isi direktori kerja Untuk masuk diirektori kerja untuk modul ini pertama adalah
   a) Jalankan command promt
   b) Masuk ke direktori kerja pada ‘C:\OS\LAB\LAB1’
   c) Cobalah untuk membuka file tersebut dengan perintah berikut, dari ‘COMMAND PROMPT’, ketikkan ‘Notepad boot.asm’



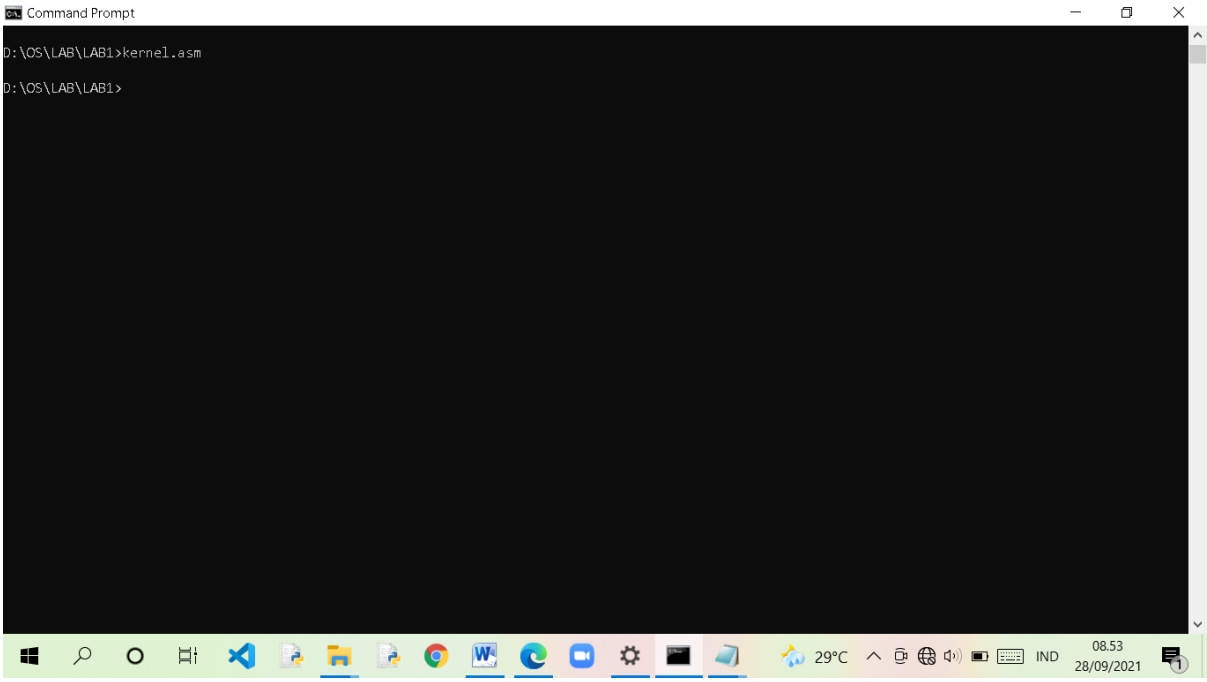


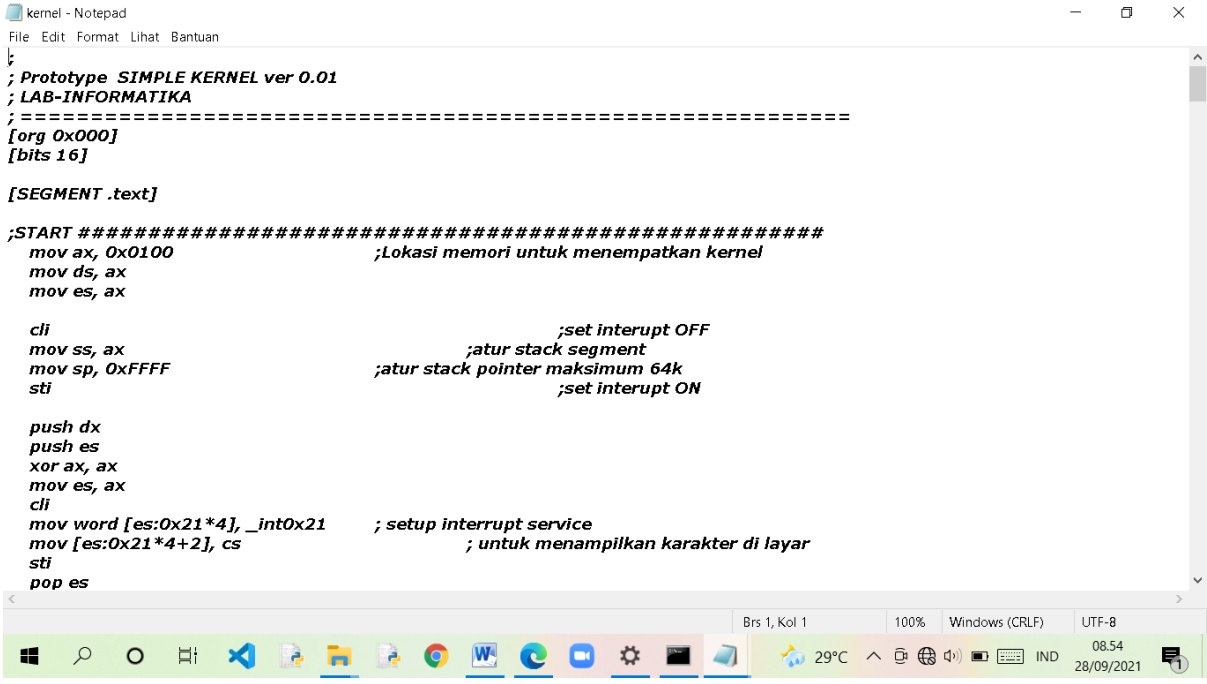
d) Source code prototype program bootloader disimpan dalam file ‘boot.asm’



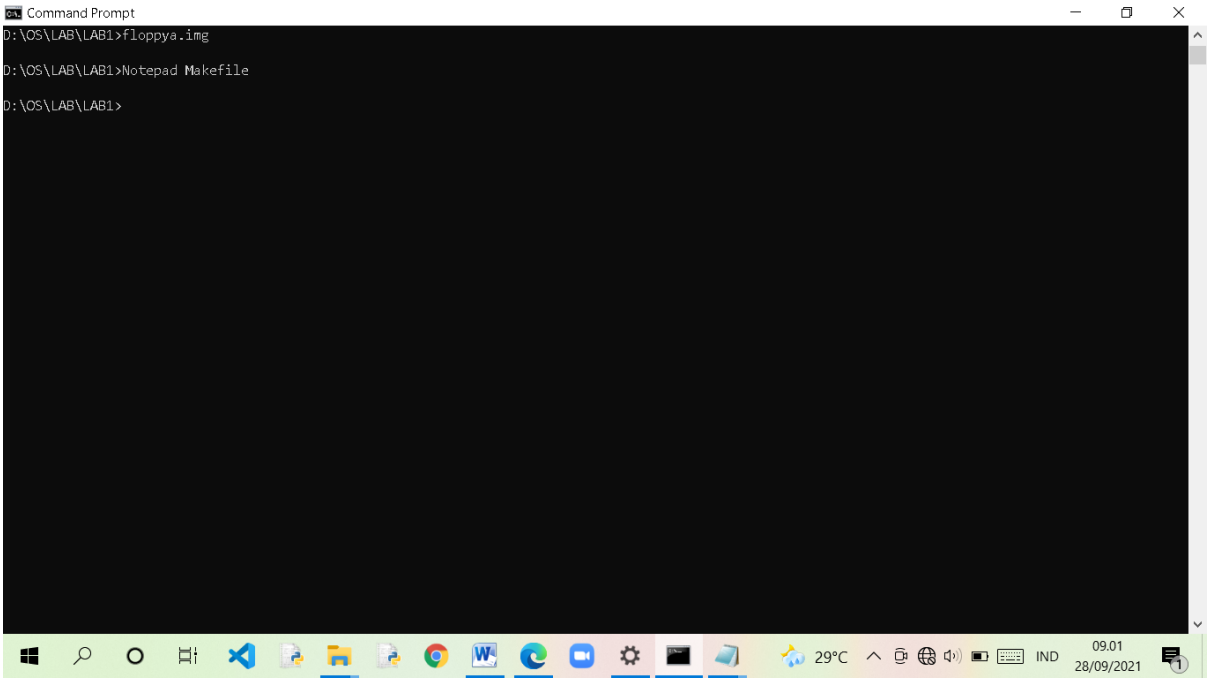


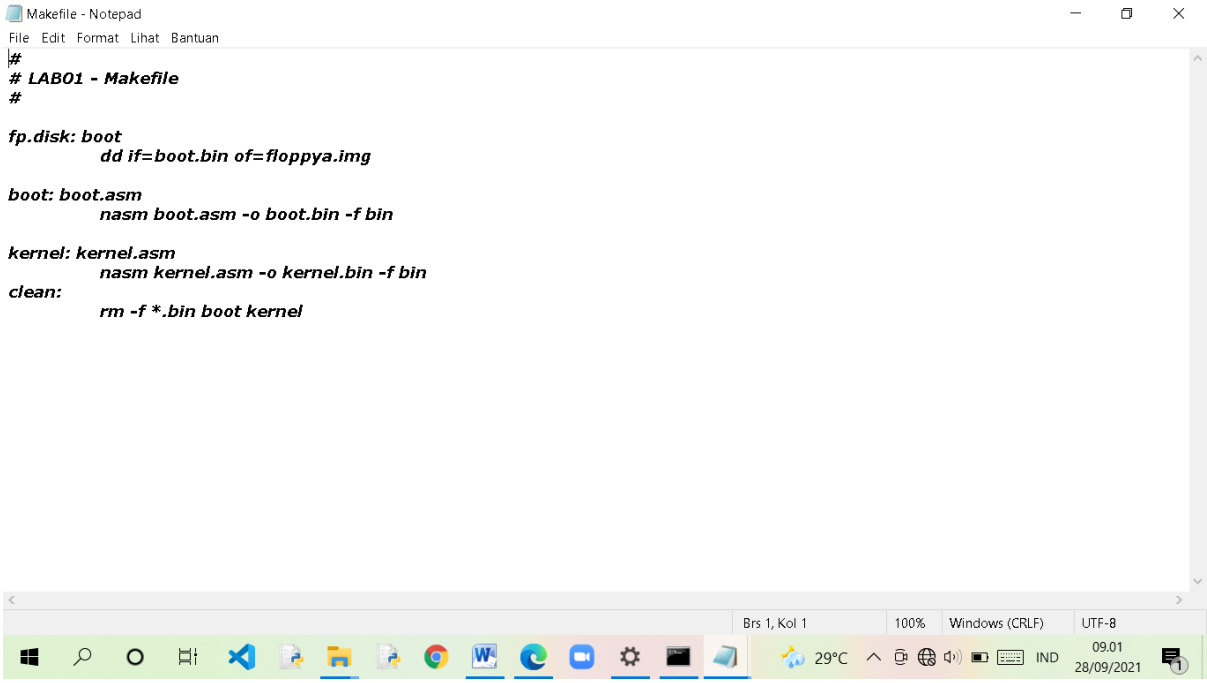
e) sedangkan untuk source code prototype program kernel disimpan dalam file ‘kernel.asm’





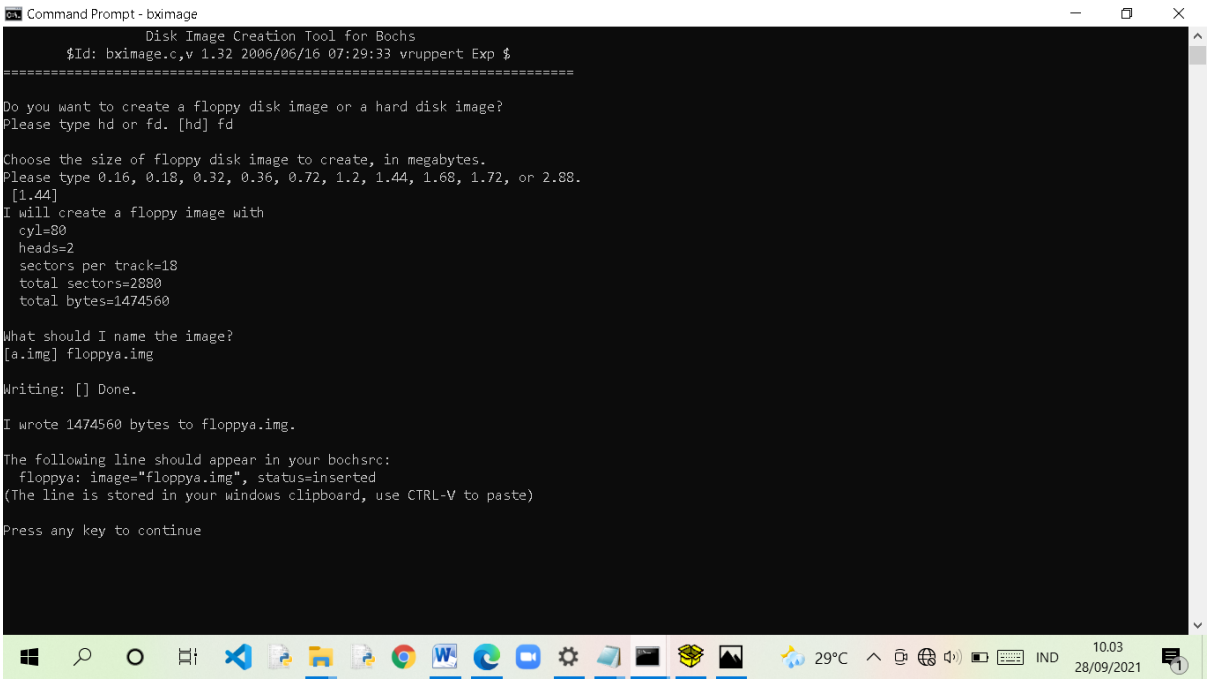
2. Makefile’ adalah sebuah file teks yang berisi kumpulan perintah ‘Command Prompt’
   a) bukalah file ‘Makefile’, dari ‘Command Prompt’ untuk mengetahui script makefile dengan cara: bukalah direktori kerja anda ‘C:\OS\LAB\LAB1’
   b) ketik ‘Notepad M’ tekan tombol ‘TAB’ sehingga muncul ‘Notepad Makefile’ dan tekan . Jika langkah anda benar akan ditampilkan windows ‘Notepad’ dengan file ‘Makefile’ yang siap disunting seperti ditampilkan pada gambar





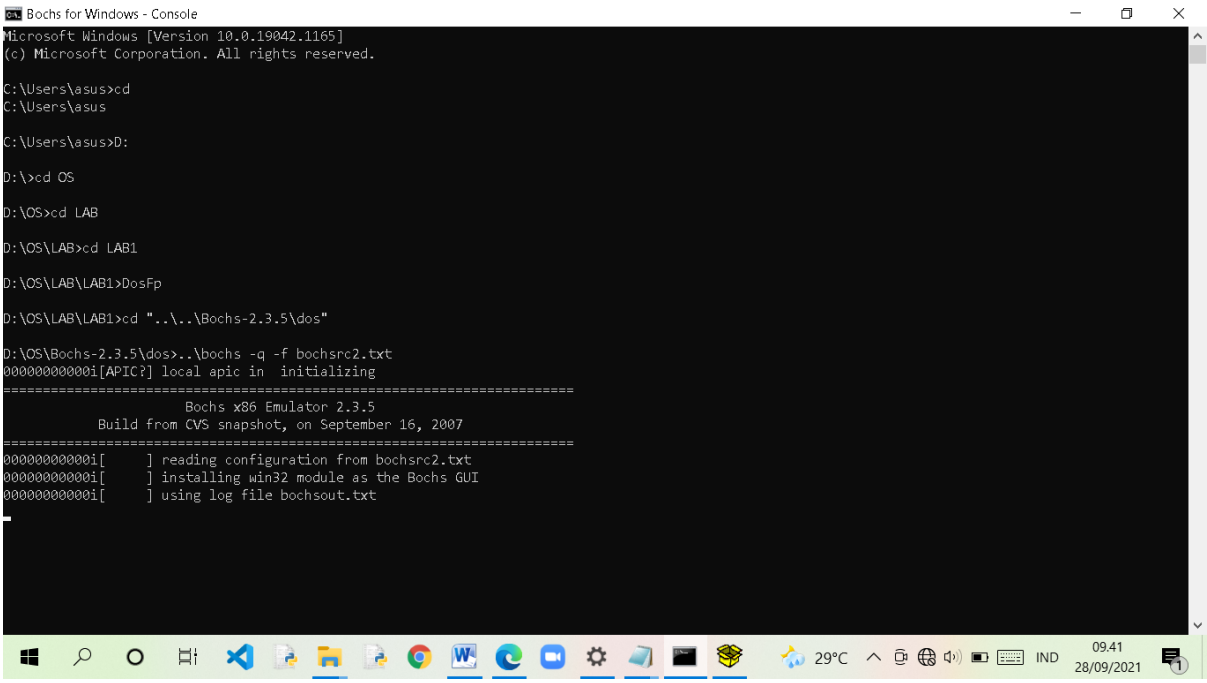
3. Mengenal ‘BOOT DISK’

a) membuat file image floppy baru dengan menggunakan program aplikasi ‘bximage.exe’,

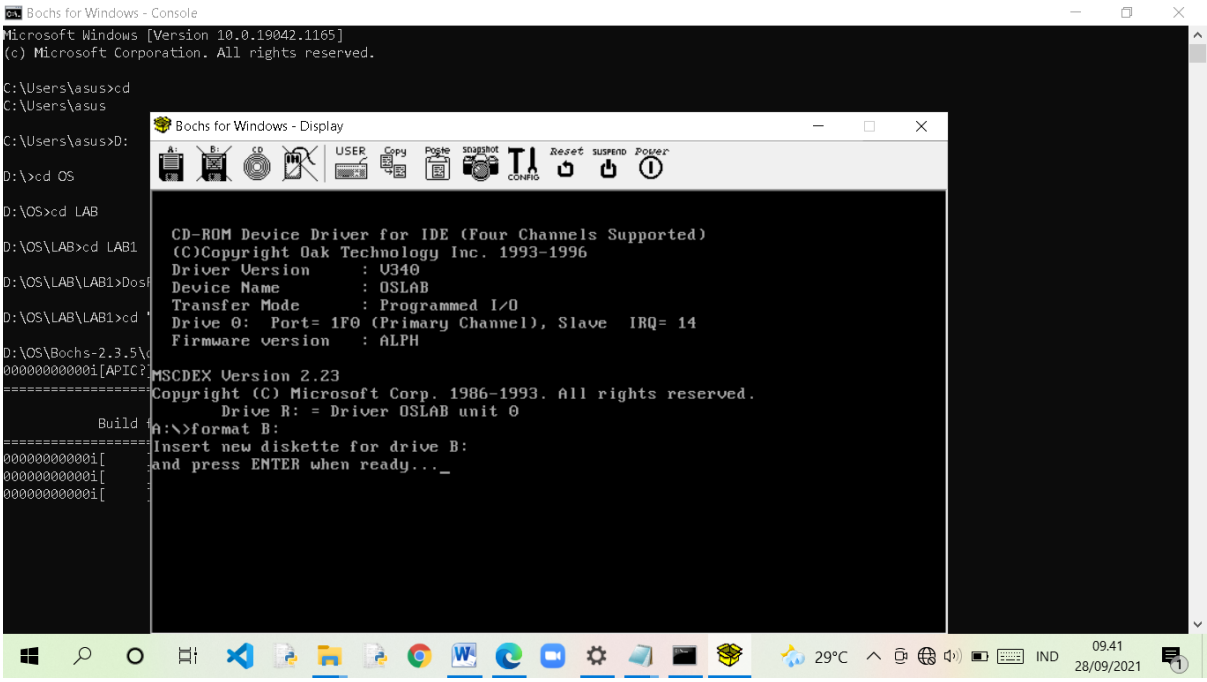


b) Langkah –langkah Untuk memformat ‘floppya.img’ adalah:

I. jalankan PC-Simulator dari ‘Command Prompt’ dengan perintah ‘DosFp’,



II. pada konfigurasi PC-Simulator file ‘floppya.img’ terpasang pada ‘drive B:’. Selanjutnya dari prompt ‘A:>’ ketikan ‘Format B:’ [2x



Tugas!!

1. **Apa yang dimaksud dengan kode ‘ASCII’, buatlah tabel kode ASCII lengkap cukup kode ASCII yang standar tidak perlu extended, tuliskan kode ASCII dalam format angka desimal, binary dan hexadesimal serta karakter dan simbol yang dikodekan.**
    - ASCII merupakan kode standar Amerika untuk pertukaran informasi atau merupakan standar internasional dalam pengkodean huruf seperti Unicode dan Hex tetapi ASCII lebih bersifat universal.
    - Tabel kode ASCII

Dec	Bin	Hex	Char	Dec	Bin	Hex	Char
0	0000 0000	00	[NUL]	32	0010 0000	20	Space
1	0000 0001	01	[SOH]	33	0010 0001	21	!
2	0000 0010	02	[STX]	34	0010 0010	22	“
3	0000 0011	03	[ETX]	35	0010 0011	23	#
4	0000 0100	04	[EOT]	36	0010 0100	24	$
5	0000 0101	05	[ENQ]	37	0010 0101	25	%
6	0000 0110	06	[ACK]	38	0010 0110	26	&
7	0000 0111	07	[BEL]	39	0010 0111	27	'
8	0000 1000	08	[BS]	40	0010 1000	28	(
9	0000 1001	09	[TAB]	41	0010 1001	29	)
10	0000 1010	0A	[LF]	42	0010 1010	2A	*
11	0000 1011	0B	[VT]	43	0010 1011	2B	+
12	0000 1100	0C	[FE]	44	0010 1100	2C	´
13	0000 1101	0D	[CR]	45	0010 1101	2D	-
14	0000 1110	0E	[SO]	46	0010 1110	2E	.
15	0000 1111	0F	[SI]	47	0010 1111	2F	/
16	0001 0000	10	[DLE]	48	0011 0000	30	0
17	0001 0001	11	[DC1]	49	0011 0001	31	1
18	0001 0010	12	[DC2]	50	0011 0010	32	2
19	0001 0011	13	[DC3]	51	0011 0011	33	3
20	0001 0100	14	[DC4]	52	0011 0100	34	4
21	0001 0101	15	[NAK]	53	0011 0101	35	5
22	0001 0110	16	[SYN]	54	0011 0110	36	6
23	0001 0111	17	[ETB]	55	0011 0111	37	7
24	0001 1000	18	[CAN]	56	0011 1000	38	8
25	0001 1001	19	[EM]	57	0011 1001	39	9
26	0001 1010	1A	[SUB]	58	0011 1010	3A	:
27	0001 1011	1B	[ESC]	59	0011 1011	3B	;
28	0001 1100	1C	[FS]	60	0011 1100	3C	<
29	0001 1101	1D	[GS]	61	0011 1101	3D	=
30	0001 1110	1E	[RS]	62	0011 1110	3E	>
31	0001 1111	1F	[US]	63	0011 1111	3F	?

Dec	Bin	Hex	Char	Dec	Bin	Hex	Char
64	0100 0000	40	@	96	0110 0000	60	`
65	0100 0001	41	A	97	0110 0001	61	a
66	0100 0010	42	B	98	0110 0010	62	b
67	0100 0011	43	C	99	0110 0011	63	c
68	0100 0100	44	D	100	0110 0100	64	d
69	0100 0101	45	E	101	0110 0101	65	e
70	0100 0110	46	F	102	0100 0110	66	f
71	0100 0111	47	G	103	0110 0111	67	g
72	0100 1000	48	H	104	0110 1000	68	h
73	0100 1001	49	I	105	0110 1001	69	i
74	0100 1010	4A	J	106	0110 1010	6A	j
75	0100 1011	4B	K	107	0110 1011	6B	k
76	0100 1100	4C	L	108	0110 1100	6C	l
77	0100 1101	4D	M	109	0110 1101	6D	m
78	0100 1110	4E	N	110	0110 1110	6E	n
79	0100 1111	4F	O	111	0110 1111	6F	o
80	0101 0000	50	P	112	0111 0000	70	p
81	0101 0001	51	Q	113	0111 0001	71	q
82	0101 0010	52	R	114	0111 0010	72	r
83	0101 0011	53	S	115	0111 0011	73	s
84	0101 0100	54	T	116	0111 0100	74	t
85	0101 0101	55	U	117	0111 0101	75	u
86	0101 0110	56	V	118	0111 0110	76	v
87	0101 0111	57	W	119	0111 0111	77	w
88	0101 1000	58	X	120	0111 1000	78	x
89	0101 1001	59	Y	121	0111 1001	79	y
90	0101 1010	5A	Z	122	0111 1010	7A	z
91	0101 1011	5B	[	123	0111 1011	7B	{
92	0101 1100	5C	\	124	0111 1100	7C	|
93	0101 1101	5D	]	125	0111 1101	7D	}
94	0101 1110	5E	^	126	0111 1110	7E	~
95	0101 1111	5F	_	127	0111 1111	7F	[DEL]

2. **Carilah daftar perintah bahasa assembly untuk mesin intel keluarga x86 lengkap (dari buku referensi atau internet). Daftar perintah ini dapat digunakan sebagai pedoman untuk memahami program 'boot.asm' dan 'kernel.asm'.**
    - ACALL (Absolute Call)
    - ADD (Add Immediate Data)

- ADDC (Add Carry Plus Immediate Data to Accumulator)
- AJMP (Absolute Jump)
- ANL (Logical AND memori ke akumulator)
- CJNE (Compare Indirect Address to Immediate Data)
- CLR (Clear Accumulator)
- CPL (Complement Accumulator)
- DA (Decimal Adjust Accumulator)
- DEC (Decrement Indirect Address)
- DIV (Divide Accumulator by B)
- DJNZ (Decrement Register And Jump Id Not Zero)
- INC (Increment Indirect Address)
- JB (Jump if Bit is Set)
- JBC (Jump if Bit Set and Clear Bit)
- JC (Jump if Carry is Set)
- JMP (Jump to sum of Accumulator and Data Pointer)
- JNB (Jump if Bit is Not Set)
- JNC (Jump if Carry Not Set)
- JNZ (Jump if Accumulator Not Zero)
- JZ (Jump if Accumulator is Zero)
- LCALL (Long Call)
- LJMP (Long Jump)
- MOV (Move From Memory)
- MOVC (Move From Codec Memory)
- MOVX (Move Accumulator to External Memory Addressed by DataPointer)
- MUL (Multiply)
- NOP (NoOperation)
- ORL (Logical OR Immediate Data to Accumulator)
- POP (Pop Stack to Memory)
- PUSH (Push Memory onto Stack)
- RET (Return from subroutine)
- RETI (Return From Interrupt)
- RL (Rotate Accumulator Left)
- RLC (Rotate Left through Carry)
- RR (Rotate Right)
- RRC (Rotate Right through Carry)
- SETB (set Carry flag)
- SJMP (Short Jump)
- SUBB (Subtract With Borrow)
- SWAP (Swap Nibbles)
- XCH (Exchange Bytes)
- XCHD (Exchange Digits)

- XRL (Exclusive OR Logic)