The following pages contain references for use during the exam: tables containing the 80386 instruction set and condition codes. You may detach these sheets from the exam and do not need to submit them when you finish.

Remember that:

- Most instructions can have at most one memory operand.
- Brackets [ ] around a register name, immediate, or combination of the two indicates an effective address. That address is in the data segment unless otherwise specified.
  - Example: MOV AX, [10H] → contents of DS:10H moved to AX
- Parentheses around a logical address mean “the contents of memory at this address”.
  - Example: (DS:10H) → the contents of memory at logical address DS:10H

Category	Instruction	Example	Meaning
Data transfer	Move	`MOV AX, BX`	`AX = BX`
	Move & sign-extend	`MOVSX EAX, DL`	`EAX = DL, sign-extended to 32 bits`
	Move and zero-extend	`MOVZX EAX, DL`	`EAX = DL, zero-extended to 32 bits`
	Exchange	`XCHG AX, BX`	`Swap contents of AX, BX`
	Load effective address	`LEA AX, [BX+SI+10H]`	`AX = BX + SI + 10H`

	Load full pointer	`LDS AX, [10H]`
		`LSS EBX, [100H]`
			`AX = (DS:10H)`
			`DS = (DS:12H)`
			`EBX = (DS:100H)`
			`SS = (DS:104H)`

Arithmetic	Add	`ADD AX, BX`	`AX = AX + BX`
	Add with carry	`ADC AX, BX`	`AX = AX + BX + CF`
	Increment	`INC [DI]`	`(DS:DI) = (DS:DI) + 1`
	Subtract	`SUB AX, [10H]`	`AX = AX – (DS:10H)`
	Subtract with borrow	`SBB AX, [10H]`	`AX = AX – (DS:10H) – CF`
	Decrement	`DEC CX`	`CX = CX – 1`
	Negate (2’s complement)	`NEG CX`	`CX = -CX`

	Unsigned multiply (all operands are non-negative, regardless of MSB value)	`MUL BH`
		`MUL CX`
		`MUL DWORD PTR [10H]`
			`AX = BH * AL`
			`(DX,AX) = CX * AX`
			`(EDX,EAX) = (DS:10H) * EAX`

	Signed multiply (all operands are signed integers in 2’s complement form)	`IMUL BH`
		`IMUL CX`
		`IMUL DWORD PTR[10H]`
			`AX = BH * AL`
			`(DX,AX) = CX * AX`
			`(EDX,EAX) = (DS:10H) * EAX`

	Unsigned divide	`DIV BH`
		`DIV CX`
		`DIV EBX`
			`AL = AX / BH (quotient)`
			`AH = AX % BH (remainder)`
			`AX = EAX / CX (quotient)`
			`DX = EAX % CX (remainder)`
			`EAX = (EDX,EAX) / EBX (Q)`
			`EDX = (EDX,EAX) % EBX (R)`

Category	Instruction	Example	Meaning
Logical	Logical AND	AND AX, BX	AX = AX & BX
	Logical inclusive OR	OR AX, BX	AX = AX | BX
	Logical exclusive OR	XOR AX, BX	AX = AX ^ BX
	Logical NOT (1's complement)	NOT AX	AX = ~AX

Shift/rotate (NOTE: for all instructions except RCL/RCR, CF = last bit shifted out)	Shift left	SHL AX, 7
		SAL AX, CX
			AX = AX << 7
			AX = AX << CX

	Logical shift right (treat value as unsigned, shift in 0s)	SHR AX, 7	AX = AX >> 7 (upper 7 bits = 0)
	Arithmetic shift right (treat value as signed; maintain sign)	SAR AX, 7	AX = AX >> 7 (upper 7 bits = MSB of original value)
	Rotate left	ROL AX, 7	AX = AX rotated left by 7 (lower 7 bits of AX = upper 7 bits of original value)
	Rotate right	ROR AX, 7	AX=AX rotated right by 7 (upper 7 bits of AX = lower 7 bits of original value)
	Rotate left through carry	RCL AX, 7	(CF,AX) rotated left by 7 (Treat CF & AX as 17-bit value with CF as MSB)
	Rotate right through carry	RCR AX, 7	(AX,CX) rotated right by 7 (Treat CF & AX as 17-b8t value with CF as LSB)
Bit test/ scan	Bit test	BT AX, 7	CF = Value of bit 7 of AX
	Bit test and reset	BTR AX, 7	CF = Value of bit 7 of AX Bit 7 of AX = 0
	Bit test and set	BTS AX, 7	CF = Value of bit 7 of AX Bit 7 of AX = 1
	Bit test and complement	BTC AX, 7	CF = Value of bit 7 of AX Bit 7 of AX is flipped
	Bit scan forward	BSF DX, AX	DX = index of first non-zero bit of AX, starting with bit 0 ZF = 0 if AX = 0, 1 otherwise
	Bit scan reverse	BSR DX, AX	DX = index of first non-zero bit of AX, starting with MSB ZF = 0 if AX = 0, 1 otherwise

Category	Instruction	Example	Meaning
Flag control	Clear carry flag	CLC	CF = 0
	Set carry flag	STC	CF = 1
	Complement carry flag	CMC	CF = ~CF
	Clear interrupt flag	CLI	IF = 0
	Set interrupt flag	STI	IF = 1
	Load AH with contents of flags register	LAHF	AH = FLAGS
	Store contents of AH in flags register	SAHF	FLAGS = AH (Updates SF,ZF,AF,PF,CF)
Conditional tests	Compare	CMP AX, BX	Subtract AX – BX Updates flags

	Byte set on condition	SETcc AH	AH = FF if condition true
			AH = 0 if condition false

Jumps and loops	Unconditional jump	JMP label	Jump to label
	Conditional jump	Jcc label	Jump to label if condition true
	Loop	LOOP label	Decrement CX; jump to label if CX != 0
	Loop if equal/zero	LOOPE label LOOPZ label	Decrement CX; jump to label if (CX != 0) && (ZF == 1)
	Loop if not equal/zero	LOOPNE label LOOPNZ label	Decrement CX; jump to label if (CX != 0) && (ZF == 0)

Condition code	Meaning	Flags
O	Overflow	OF = 1
NO	No overflow	OF = 0
B NAE C	Below Not above or equal Carry	CF = 1
NB AE NC	Not below Above or equal No carry	CF = 0
S	Sign set	SF = 1
NS	Sign not set	SF = 0
P PE	Parity Parity even	PF = 1
NP PO	No parity Parity odd	PF = 0
E Z	Equal Zero	ZF = 1
NE NZ	Not equal Not zero	ZF = 0
BE NA	Below or equal Not above	CF OR ZF = 1
NBE A	Not below or equal Above	CF OR ZF = 0
L NGE	Less than Not greater than or equal	SF XOR OF = 1
NL GE	Not less than Greater than or equal	SF XOR OF = 0
LE NG	Less than or equal Not greater than	(SF XOR OF) OR ZF = 1
NLE G	Not less than or equal Greater than	(SF XOR OF) OR ZF = 0