NAME : ADEROJU OLUWAFIKEMI ADEBOSOLA
DEPARTMENT : COMPUTER ENGINEERING
MATRIC NO: 18/ENG02/102 fikemA

Question 1.

a) It will not be good because the coded addresses in the instructions would have to be updated when new variable are inserted before existing ones

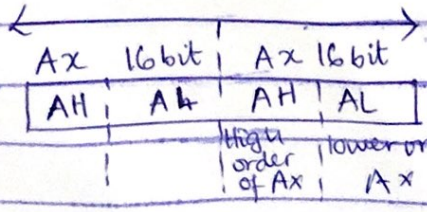
b) An Object File

Question 2

a) Portability in programming language is the ability of an application to run properly on a different platform to the one it was originally designed for without need for modification.

b) No, it is not. This is because each assembly language is based on either a processor family or a specific computer

c) The EX register is known as Accumulator because it is the destination for many arithmetic results. Given an instruction EAX 200. The instruction above will add the decimal number, convert to double word length to the number in the Eax register which replaces the original number with the sum of the two numbers.



ADEROJU FIKEMI
18/ENG02/102

Question 3

3a.) Segmentation in Assembly language is achieved by dividing the system memory into groups of independent segments which are referenced by pointers located in the segment registers.

3b.)
```
Main PROC
    Mov AX, 47104;
    ADDEAX, 127Q
    MOVDS, Ax
     main ENDP
```
Main PROC means the starting of the procedure
mov AX, 47104 means that the computer should copy the number 47104 to the location AX
ADDEAX, 127Q means the computer should add 127Q to EAX register.
MOV DS, AX means to copy the number in the AX register in to the DS register location
main END P means the end of the procedure.

3c.) Value 1 BYTE 6DH
→ This means that 6DH is a hexadecimal byte
ii.) Value 2 DWORD?
→ This means that it is a double word
iii.) Value 3 SBYTE -10, -20, -30, -40, -50
→ This means that this code becomes a variable called "value 3" which has the equivalent of a signed byte

ADEROJU FIKEMI 18/ENG02/102 ~~FIKEMI~~

Question 4

4.) TITLE : Subtract 3 integers using only 16 bits registers

```
INCLUDE Irvine32.Inc
.code
 main PROC
mov eax, 5000h
sub eax, 3000h
sub eax, 2000h
Call Dump Regs
exit
main ENDP
```