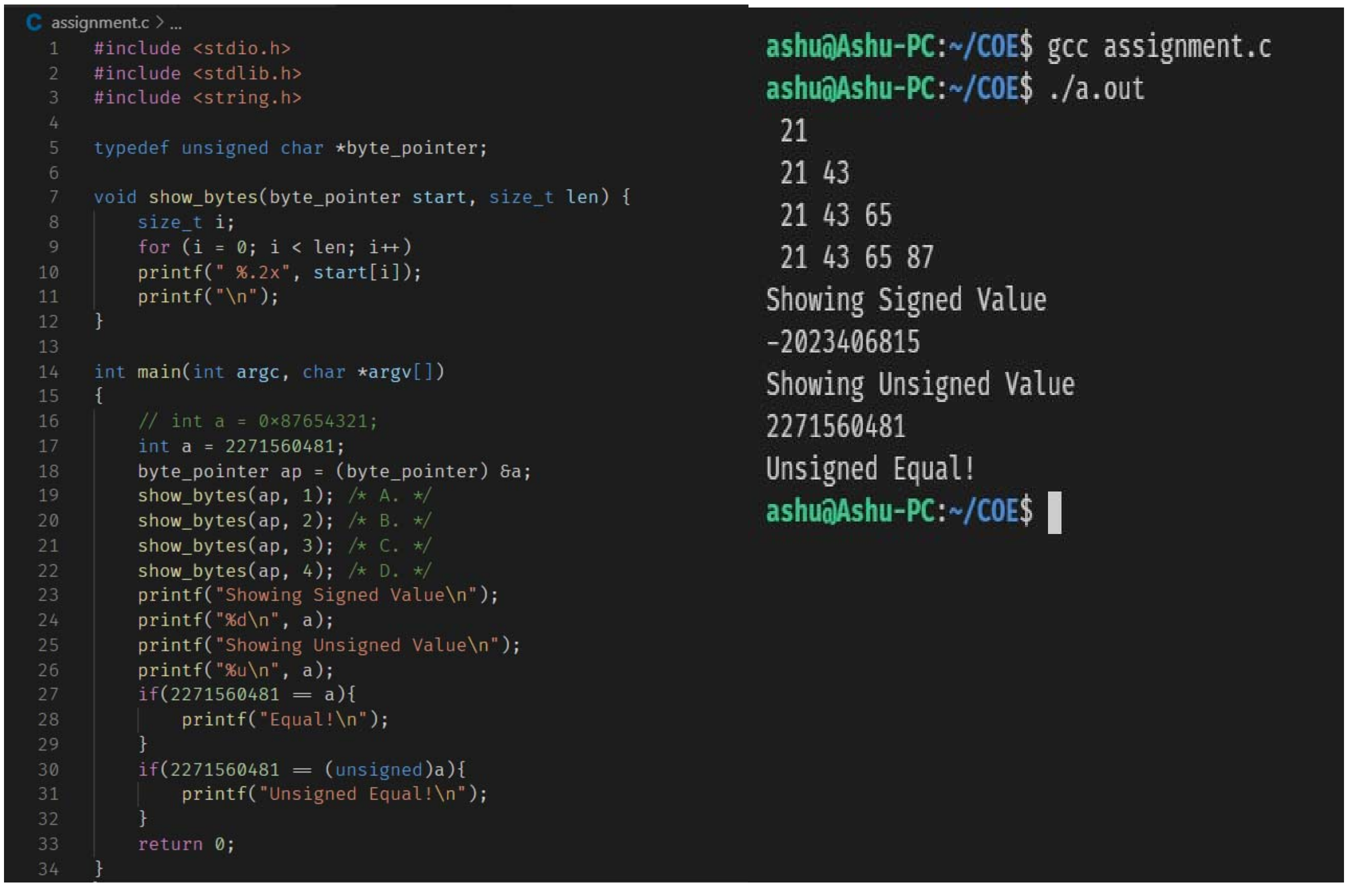
Task 4:

How does the Hexadecimal representation of 0x87654321 after getting stored as an integer retain correct value as hexadecimal even if the decimal representation is out of the range of the standard integer datatype in C.



assignment.c > ...

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef unsigned char *byte_pointer;

void show_bytes(byte_pointer start, size_t len) {
    size_t i;
    for (i = 0; i < len; i++)
    printf(" %.2x", start[i]);
    printf("\n");
}

int main(int argc, char *argv[])
{
    // int a = 0x87654321;
    int a = 2271560481;
    byte_pointer ap = (byte_pointer) &a;
    show_bytes(ap, 1); /* A. */
    show_bytes(ap, 2); /* B. */
    show_bytes(ap, 3); /* C. */
    show_bytes(ap, 4); /* D. */
    printf("Showing Signed Value\n");
    printf("%d\n", a);
    printf("Showing Unsigned Value\n");
    printf("%u\n", a);
    if(2271560481 == a){
        printf("Equal!\n");
    }
    if(2271560481 == (unsigned)a){
        printf("Unsigned Equal!\n");
    }
    return 0;
}
```

```
ashu@Ashu-PC:~/COE$ gcc assignment.c
ashu@Ashu-PC:~/COE$ ./a.out
 21
 21 43
 21 43 65
 21 43 65 87
Showing Signed Value
-2023406815
Showing Unsigned Value
2271560481
Unsigned Equal!
ashu@Ashu-PC:~/COE$
```

The Given Number will be converted to Binary then stored in the memory starting with Least Significant Bit , it continues to fill all the space allocated discarding any Higher Bits, if any.

For The number 2271560481 or 0x87654321 The No of bits in binary will be 32 bits.

0x87654321 = $(1000\ 0111\ \ 0110\ 0101\ \ 0100\ 0011\ \ 0010\ 0001)_2$

And Stored as (considering little endian):

0x00000000	0x00000001	0x00000002	0x00000003
0 0 1 0 0 0 0 1	0 1 0 0 0 0 1 1	0 1 1 0 0 1 0 1	**1** 0 0 0 0 1 1 1
0x21	0x43	0x65	0x87

Since the value is saved as int the MSB **(BOLD)** is used to define sign, and since MSB is ‘**1**’ it will be considered a negative number and value will calculated using two’s complement which will be ‘***-2023406815’***.

But we print the output Byte by Byte using an *unsigned char** we get positive numbers and hence get correct representations, when we print the output.