

# Assignment 6

Q1. WAP to Check the endianness (little or big) of your current system.

Q2. WAP to Conversion of short integer from little endian to big endian(network order) and vice versa.

Q3. WAP Conversion of integer from little endian to big endian(network order) and vice versa.

Q4. Given `int a[5] = {10, 20, 30, 40, 50};`  
`int *p=a, *q=(&a+1) - 1;`  
evaluate following expressions  
`*p++`, `*++p`, `(*p)++`, `++(*p)`, `++*p`, `*(p++)`, `*(++p)`  
`*q--`, `*--q`, `--(*q)`, `--*q`,  
`(*q)--`, `*(q--)`, `*(--q)`

Q5. WAP to Convert from one type of pointer/address to other using `void*`

Q6. WAP to Test arithmetic operation on void pointers

Q7. `int arr[5]; int (*parr)[5];`  
`parr=&arr;`  
`sizeof(parr)`, `sizeof(*parr)`, `sizeof(**parr)`  
access array elements with suitable dereferencing of `parr`

Q8. Usage of `assert` macro before dereferencing any pointer.

Q9. Differentiate between  
`const int * p;`  
`int const * p`  
`int * const p = &x;`  
`const int * const p = &x;`  
If `*p=20`, `p++`, `(*p)++`, `p=&y` in each case

Q10. Access 2D array using pointers  
`int arr[3][4]; int (*p)[4]; p=arr;`  
`sizeof(p)`, `sizeof(*p)`, `sizeof(**p)`, values of `p`, `p+1`  
Check equivalence of `arr[i][j]`, `*(p+i)[j]`, `*(*(p+i)+j)`

Q11. Give an expression to print last element of array irrespective of length using pointer notation.  
(You shouldn't consider length or size of array)

Q12. What is the significance of following pointer  
`int (*q)[3][4];`

What are the sizes of `q`, `*q`, `**q`, `***q`

Write some code to test this with a 2D array.