Assignment-9

Q] What is meant by NULL pointer? & Why to initialise uninitialised pointer to NULL?

Ans] When we create any type of pointer & if that pointer is uninitialized then it may contain garbage values in it

] Due to unwanted garbage value it may lead to run-time failure (run-time accidents)

] To avoid this we use the concept of NULL pointer.

] NULL is a macro which is defined in <stdio.h> header file.

] the value of that macro is '0' (zero)

] When we initialise any uninitialized pointer with NULL it helps to reduce the run-time accidents (segmentation faults)

] It is a good programming practice to initialise any pointer with NULL.

Segmentation Fault

] When a process tries to access the contents which are outside its address space then the OS will kill the running process by giving the segmentation fault.

] In any operating system, it is not allowed to access contents which are outside the address space of a process.

Address space

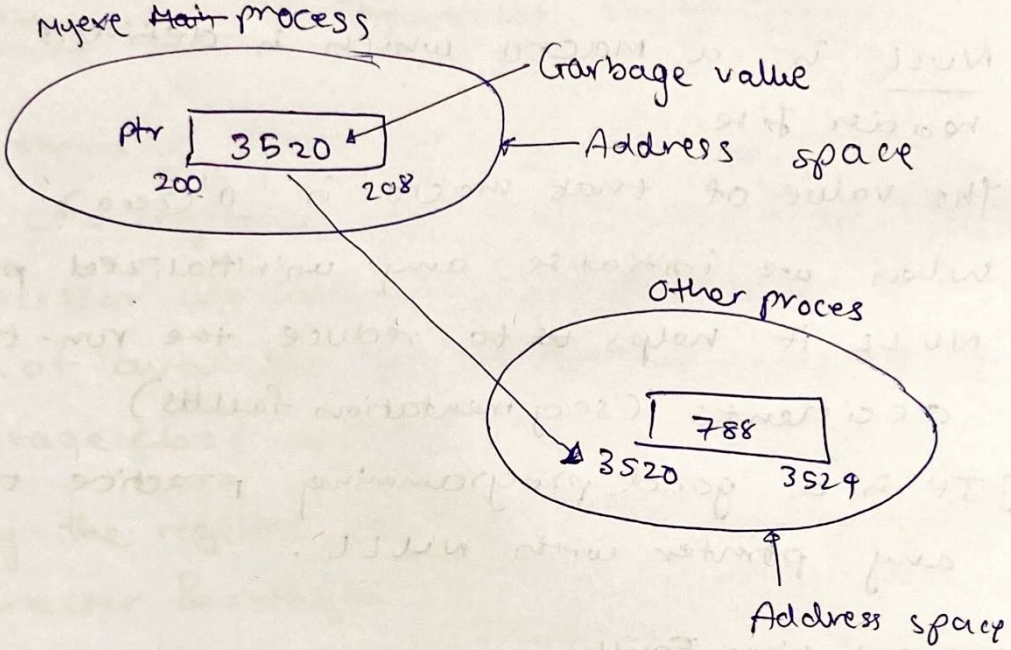
] When we execute any program it is considered as a process.

] When any process gets executed the OS will allocate some amount of memory to that process & that memory is called as Address space.

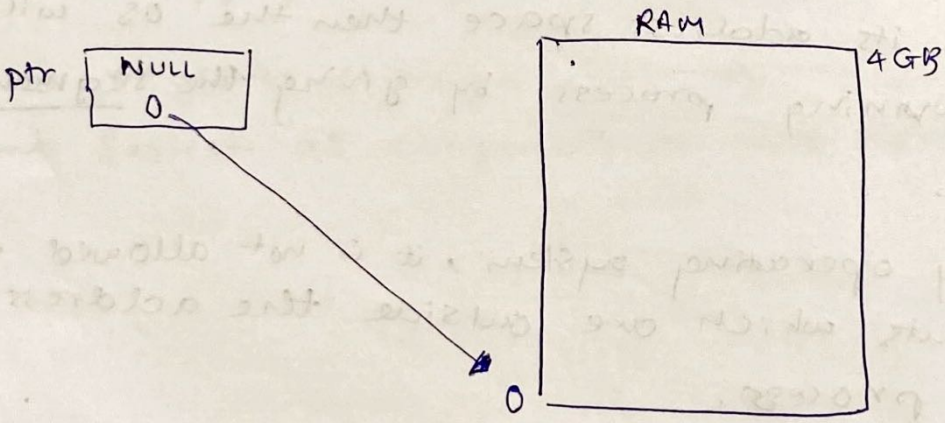
consider the following scenario, which leads to Segmentation fault.

```
int main()
{
    int *ptr;
    printf("%d", *ptr);
}
```

Layout



To avoid such kind of process, we initialise pointer to NULL.

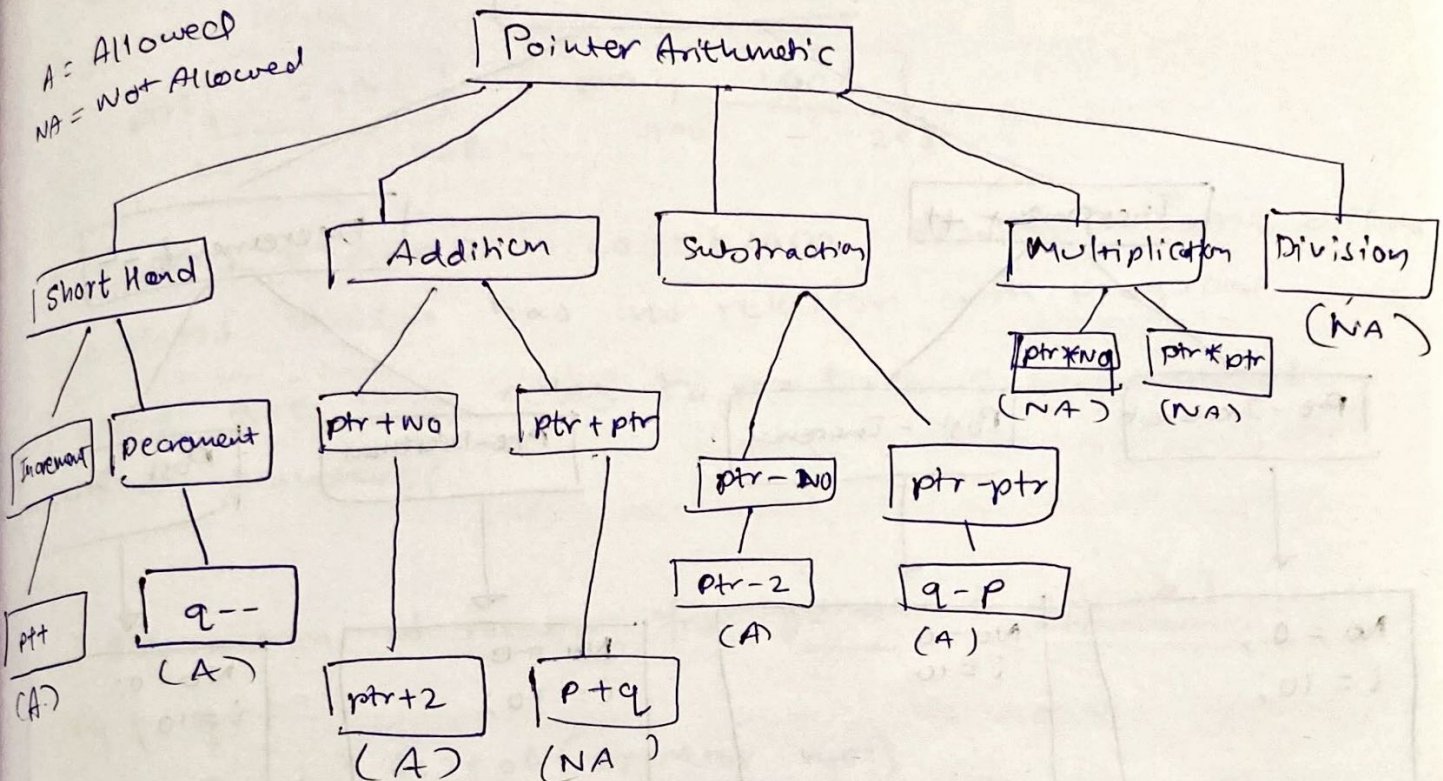


- In *ptr = NULL, NULL is internally defined as #define NULL (void*)0.
- AS NULL is Macro, after every preprocessing every occurence of NULL gets replaced with 0.
- The zero address is bottom of our RAM & there is no address space of any proces.

Q2] What is meant by pointer arithmetic? Explain in details?

Ans:] Pointer arithmetic are not normal arithmetic operations like of numbers.

] There are 5 arithmetic operations like additions, subtraction, multiplication, division, short hand operators.



] Both pointers should be of same data type.

] Both pointer should point to the same memory region (expected array)

Q3] What are the tasks of operating System?

Ans 1] File management

2] Process management

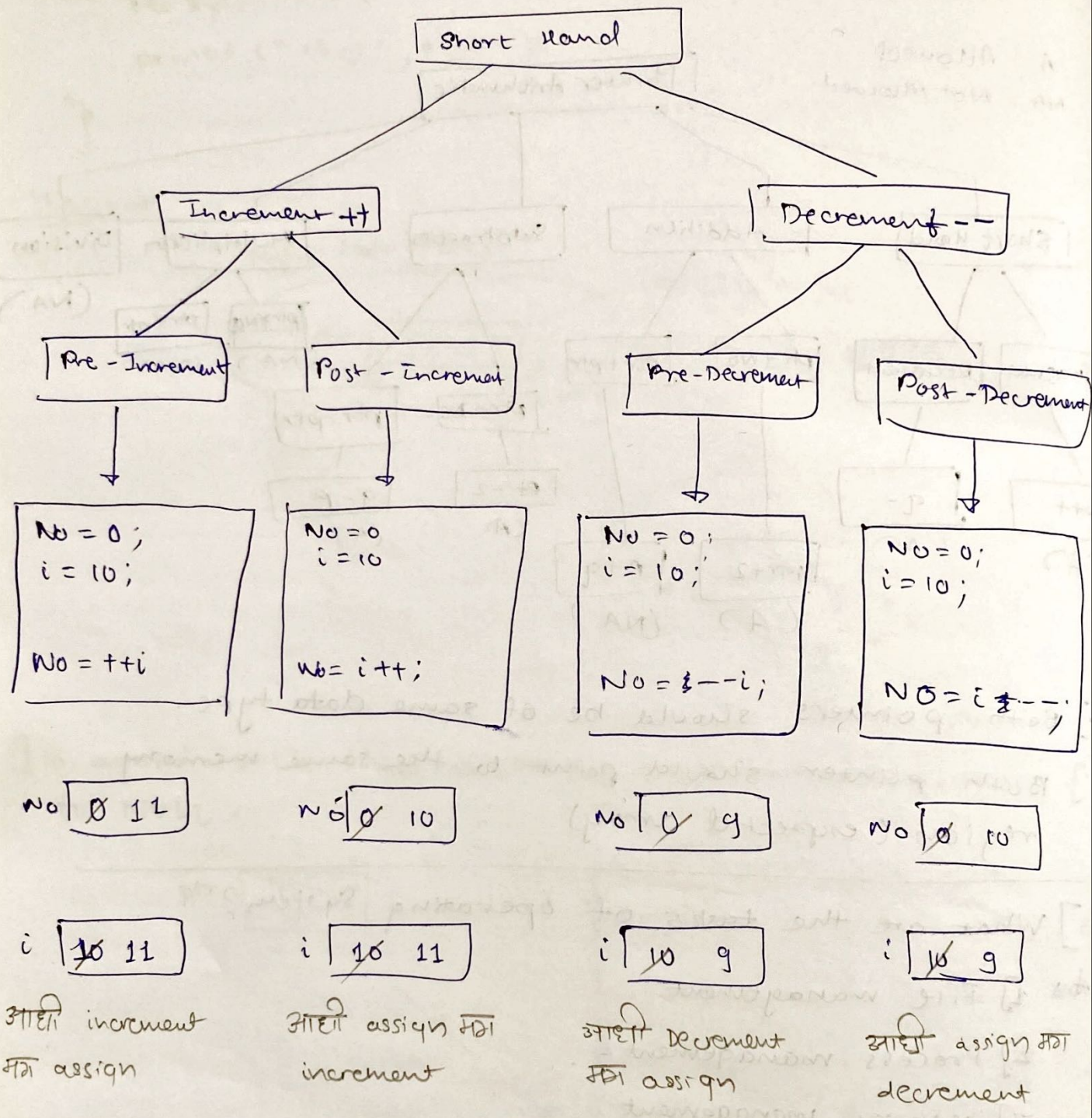
3] Memory management

4] CPU scheduling

5] Hardware abstraction

Q] What is meant by increment & decrement operator?

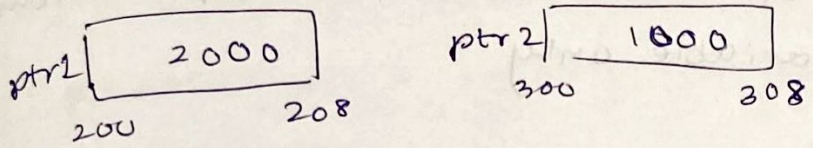
Ans In C, C++ & Java there are two operators which are used to increment the value by 1 & decrement the value by 1



:] If increment operator is used independently it is normal increment & decrement and not pre / post decrement or increment.

Q5] why we cannot perform addition of two pointers?

Ans Since pointers store address, two addresses & their addition might result of some other address of which may be outside of address space



ptr1 + ptr2 = 2000 + 1000 results in some different address which has no relation with program

Q6] What are the rules to perform subtraction of two pointers?

Ans

1] Subtraction of number from pointer is allowed

Eg: q - 2;

= q - 2 * sizeof(operator-type)

= q - 2 * sizeof(int)

= q - 8 2*4 = q - 8 = 116 - 8

= 108

2] Subtraction of pointer from number is not allowed

Eg: 2 - q; Not allowed

3] Subtraction of pointer from pointer is allowed.

It is only meaningful if both pointers point to the same arrays elements. their difference is the difference bet two array elements

q - p;

= (q - p) / sizeof(pointer type)

= (q - p) / int

= (116 - 108) / 4 => 2 No of elements bet them.

Q7] What is Difference bet Declaration, Definition & Initialisation?

Ans * Declaration

:) Declaration is considered as a place where there is no memory allocation
:) At the point of declaration the compiler considers the name of variable only.

~~* Initialisation~~

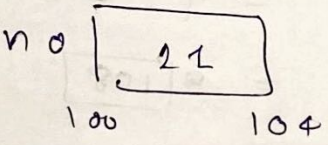
-> * Definition

:) Definition is concept where memory gets allocated to the variable as well as name of variable gets registered by the compiler.

* Initialisation

:) Initialisation is a place where memory gets allocated to the variable as well as the value of that variable is initialized.

Eg: int no = 12;



In above example compile registers the name no, allocates memory to variable & initializes value to it

Q8] Predict the output

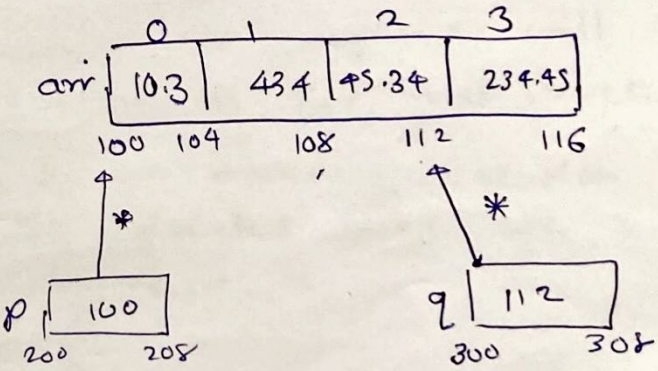
```
int main ()
{   int no = 10;
    int *p = NULL
    p = &no;
```

8] Predict the output

```
#include <stdio.h>
int main()
{
    int no = 10;
    int *p = NULL;
    p = &no;

    printf("%d", no);        // 10
    printf("%d", *p);        // 10

    *p = 11;

    printf("%d", no);        // 11
    printf("%d", *p);        // 11

    return 0;
}
```

9] Predict the output

```
#include <stdio.h>
int main()
{
    float arr[] = {10.3, 43.4, 45.34, 234.45};
    float *p = NULL;
    float *q = NULL;

    p = arr;
    q = &(arr[3]);

    printf("%d", p);        // 100
    printf("%d", q);        // 112
    printf("%f", *q);       // 234.45
    printf("%f", *p);       // 10.3
    printf("%f", *(p+2));   // 45.34
    printf("%f", *(p+1));   // 43.4
    printf("%f", p[1]);     // 43.4
    printf("%f", *(2+arr));// 45.34
```



```
printf("%f", 0[arr]); //10.3
printf("%d", (q-p)); // 3
printf("%f", *(q-2)); //43.4
return 0;
}
```

Q10] Explain how array is considered as pointer & pointer can be treated as array?

Ans :] The name of an array refers to the address of first element of the array

:] Since it refers to an address (it becomes pointer type.

:] arr[1] is automatically converted to *(arr + 1)

:] array[0], *(arr+0), 0[array] are considere same.