Date 27/11/2023 Page

Day - 64

Pointers in C++ - 2

* Suppose, now we have a array —

int arr[5]

0	1	2	3	4
11	7	8	12	14
500	504	508	512	516

int *ptr = &arr[5];

⇒ We know that, arr store the address of first element.

So,

arr	arr+1	arr+2	arr+3	arr+4
500	504	508	512	516

⇒ arr+2 →

i^{th} index add^n = Base add. + i X data type size

arr + 2 = 500 + 2 X 4 = 508

⇒ So, arr[i] = *(arr+i) = *(i+arr)

↓ ↗

i[arr]

Also,

arr = &arr[0]

= & *(arr+0)

= & * (arr) ('&' & '*' cancel each other)

arr = arr

Date ______ Page ______

```
// Print the values by using *(arr+i)
for (i=0; i<5; i++)
        cout<< *(arr+i) <<endl;

// Print the addresses
for( i=0; i<5; i++)
        cout<< (arr+i);
        cout<< endl;
```

=> We can also print values by using ptr.

```
for (i=0; i<5; i++)
        cout << ptr[i];
        cout << *(ptr+i);
```

(++, --)

=> 'ptr' can also do arithmetic operations -

```
for(i=0; i<5; i++){
        cout<<*ptr;
        ptr++;
}
```

=> We can't perform arr++ & arr--.

=> Anything that goes into the symbol table, we can't change it.

=> In the same way, we can't change the address of ptr, we only change the value of ptr.