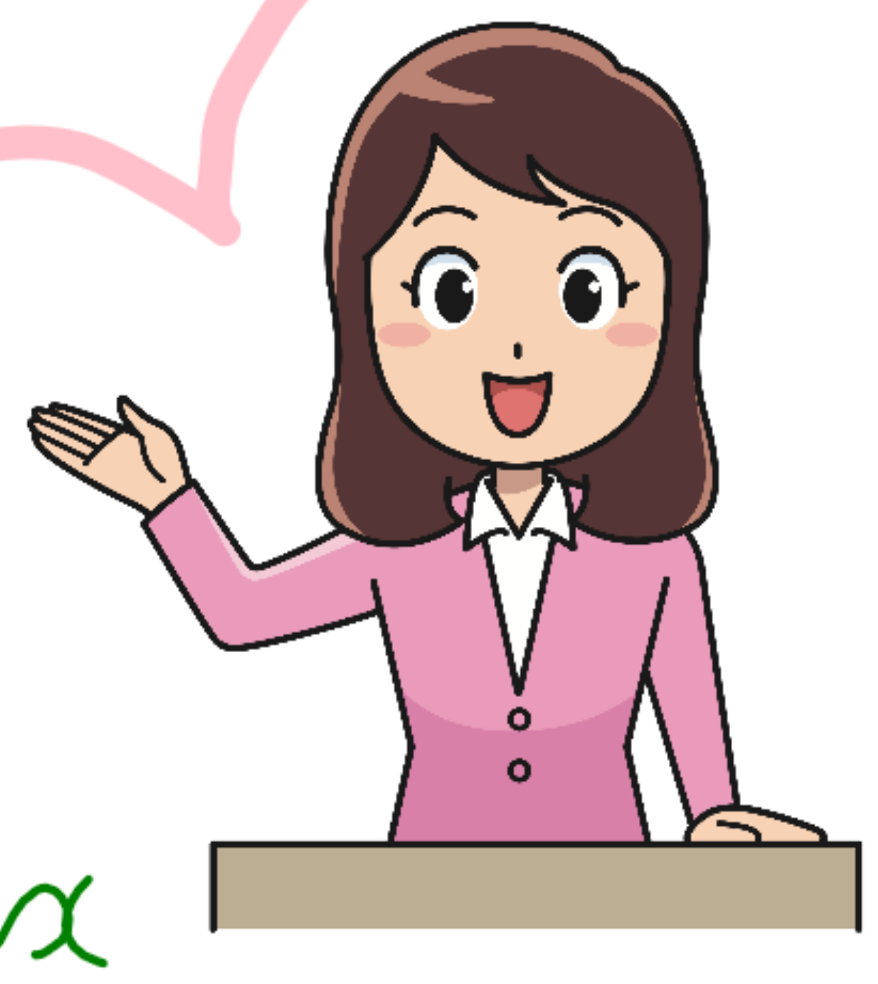
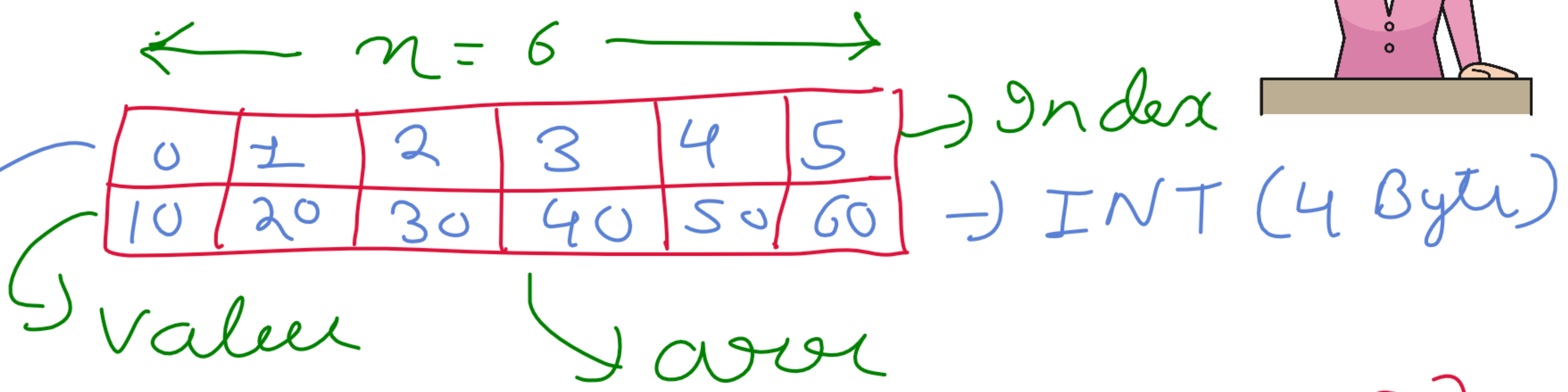
1D - Array

Introduction to Array

An array can be defined as a linear data structure, which is used to store a sequence of values of the same type.





Index ∈ [0, n-1], here [0, 5]

→ Total Size = Size of INT * count of arr[]
= 24 byte

```
#include<iostream>
using namespace std;

int main(){

    int arr[5];
    cout << endl << sizeof(arr) << endl;

    char str[5];
    cout << sizeof(str) << endl;

    double bbr[5];
    cout << sizeof(bbr) << endl;

    //assigning value to arr
    for(int i = 1; i <= 5; i++)
    {
        arr[i] = i;
        cout << arr[i] << " ";
    }

    //size of array
    int size = sizeof(arr)/sizeof(int);
    cout << endl << size << endl;

    //array initialization
    int array[5] = {1,2,3,4,5};
```

→ array initialization (with garbage value)

→ assigning values

→ calculating size of arr

→ array initialized with values

Array Initialization

Method - I

```
//array initialization
int array[5] = {1,2,3,4,5}; //size setting is optional here
for(int i = 0; i < 5; i++)
{
    cout << endl << array[i] << " ";
}
//int array[5] = {1,2,3,4,5,6}; ERROR: NOT ALLOWED EXCCEDING THE ARRAY SIZE
```

→ array initialized with 1 to 5.

→ will give error.

```
int A[5] = {1,2,3}; //ZERO WILL BE FILLED
cout << endl << "1." << endl;
for(int i = 0; i < 5; i++)
{
    cout << A[i] << " ";
}

int B[5] = {0}; //ZERO WILL BE FILLED
cout << endl << "2." << endl;
for(int i = 0; i < 5; i++)
{
    cout << B[i] << " ";
}
```

→ If not given the, Initial value 0 will be set (no garbage value).

Method - II

```
int G[5];
memset(G,-1,sizeof(F));
cout << endl << "5." << endl;
for(int i = 0; i < 5; i++)
{
    cout  << G[i] << " ";
}
```

→ (cstring-lib) memset function can be used to initialize value

memset(arr, value, size of arr)

take three parameters

```
int H[5];
memset(H,2,sizeof(F)); //memset can only be used to init arr with 0 or -1
cout << endl << "6." << endl;
for(int i = 0; i < 5; i++)
{
    cout  << H[i] << " ";
}
```

↓ memset limitation

It is due to, memset saving in 8-bits × 4 to form 32-bit Int.

$[00000010\ 00000010\ 00000010\ 00000010]_2 = (33686018)_{10}$

↓

This number will be stored.

Reading input to array (Saving values)

```
int Z[5]; //to fill 2
cout << endl << "ENTER NUMBERS:." << endl;
for(int i = 0; i < 5; i++)
{
    cin >> Z[i];
}
cout << endl << "INPUT TAKEN." << endl;
cout << endl << "Printing." << endl;
for(int i = 0; i < 5; i++)
{
    cout << Z[i] << " ";
}
cout << endl << "Reverse Printing." << endl;
for(int i = 4; i >= 0; i--)
{
    cout << Z[i] << " ";
}
```

} Reading values and saving them, through loop (itr -> index)

]- Printing array

]- Reverse printing arr (itr in reverse)

Array (1-D) Based example questions

First Occurence [To find a target value and return its index of first occurence)

Code:-

(Given)

```
int arr[100] = {1,2,3,4,5}; //n<=100 constrain
int size = sizeof(arr)/sizeof(int);
int index = -1;
int target = 1;

//linear search
//worst case will be when target not found, O(n)

for(int j=0; j<size ;j++)
{
    if(arr[j]==target)
    {
        index = j;
        cout << "\nFirst Occurence is at index: " << j;
        break;
    }
}
if (index==-1)
{
    cout << "\nTarget not found.";
}
```

↳ calc size(arr)

↳ linear Search will be used

↳ To find the target.

↳ To find only one occurence

} If target is not found (condition)