

Array - Collection of data storage

Storage location

- A data structure that holds Collection of Elements of the same data type.
- Elements are stored in contiguous memory location.
↳ sequential / pattern

Declaring Array / single dimension Array

- like any Variable an array has to be declared before use as :-

Data Type Array Name [Size of the Array]

Eg. Declare arrays that contains ages and Scores of 5 students

age 0 1 2 3 4
 23 19 21 20 19

age[0] = 23
age[1] = 19
age[5] = no

→ int age [5];
→ float score [5];

Accessing Elements of an Array

- to access the element of an array we use an index, which ranges from 0 to N-1, where N is # of elements in the array.

⇒ To traverse / visit all elements of an array we use iterative statements

Write a C++ program that accepts scores of five students & stored it in an array name stud-score. Then your program should print the content of the array.


```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
main()
```

```
{ float stud_score[5]; // Array declaration
```

```
// To fill the array
```

```
for (int k=0; k<5; k++)
```

```
{ cout << "Enter score";
```

```
cin >> stud_score[k];
```

```
// To print contents of the array
```

```
for (int i=0; i<5; i++)
```

```
cout << stud_score[i] << " ";
```

```
}
```

#. Write a C++ program that accepts 10 integers/determines & prints to an array named myArray, & determine & prints the maximum element

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
main()
```

```
{ const int SIZE = 10;
```

```
int myArray[SIZE]; // Array decln, max = 0;
```

```
for (int i=0; i<SIZE; i++)
```

```
{ cout << "Enter element " << i << ": -";
```

```
cin >> myArray[i];
```

```
if (myArray[i] > max)
```

```
max = myArray[i];
```

} To find max^m element

```
{ cout << "Maximum element is " << max << endl;
```

```
}
```

MY COLLEAGUE


```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
main()
```

```
{ int ar[10];
```

```
for (int k=0; k<10; k++)
```

```
{ ar[k] = k * 3 + 3;
```

```
if (k % 2 == 0 || k % 3 == 0)
```

```
ar[k] -= 2 * k;
```

```
}
```

0	1	2	3	4	5	6	7	8	9
3	5	3	3	3	13	3	17	3	3

```
for (int k=0; k<10; k++)
```

```
cout << ar[k] << " ";
```

```
}
```

Initializing An Array

size accommodate the element

- fill the array with elements (automatically)

NOTE-We can initialize an array as:-

Data Type Array Name [size] = { list of comma separated elements };

Eg, initialize ages of 5 students

```
int age[5] = { 20, 19, 21, 23, 18 }; [3] x
```

2) Initializing names of 5 students

[10] ✓

```
String names[5] = { "aa", "cc", "bb", "ee", "dd" };
```

6 - 10

ar[0] = 3

ar[1] = 3 - 2 = 1

ar[2] = 6

ar[3] = 7

ar[4] = 7 - 4 = 3

ar[5] = 9

ar[6] = 9 - 6 = 3

ar[7] = 11

ar[8] = 11 - 8 = 3

ar[9] = 3

ar[10] = 5

ar[11] = 5 - 2 = 3

ar[12] = 7

ar[13] = 9

ar[14] = 9 - 16 = -7

ar[15] = 18 + 3 = 21

ar[16] = 21 - 18 = 3

ar[17] = 18 + 3 = 21

ar[18] = 21 - 18 = 3

ar[19] = 21 - 18 = 3

ar[20] = 21 - 18 = 3

ar[21] = 21 - 18 = 3

ar[22] = 21 - 18 = 3

ar[23] = 21 - 18 = 3

ar[24] = 21 - 18 = 3

ar[25] = 21 - 18 = 3

ar[26] = 21 - 18 = 3

ar[27] = 21 - 18 = 3

ar[28] = 21 - 18 = 3

ar[29] = 21 - 18 = 3

ar[30] = 21 - 18 = 3

ar[31] = 21 - 18 = 3

ar[32] = 21 - 18 = 3

ar[33] = 21 - 18 = 3

ar[34] = 21 - 18 = 3

ar[35] = 21 - 18 = 3

ar[36] = 21 - 18 = 3

ar[37] = 21 - 18 = 3

ar[38] = 21 - 18 = 3

ar[39] = 21 - 18 = 3

ar[40] = 21 - 18 = 3

ar[41] = 21 - 18 = 3

ar[42] = 21 - 18 = 3

ar[43] = 21 - 18 = 3

ar[44] = 21 - 18 = 3

ar[45] = 21 - 18 = 3

ar[46] = 21 - 18 = 3

ar[47] = 21 - 18 = 3

ar[48] = 21 - 18 = 3

ar[49] = 21 - 18 = 3

ar[50] = 21 - 18 = 3

ar[51] = 21 - 18 = 3

ar[52] = 21 - 18 = 3

ar[53] = 21 - 18 = 3

ar[54] = 21 - 18 = 3

ar[55] = 21 - 18 = 3

ar[56] = 21 - 18 = 3

ar[57] = 21 - 18 = 3

ar[58] = 21 - 18 = 3

ar[59] = 21 - 18 = 3

ar[60] = 21 - 18 = 3

ar[61] = 21 - 18 = 3

ar[62] = 21 - 18 = 3

ar[63] = 21 - 18 = 3

ar[64] = 21 - 18 = 3

ar[65] = 21 - 18 = 3

ar[66] = 21 - 18 = 3

ar[67] = 21 - 18 = 3

ar[68] = 21 - 18 = 3

ar[69] = 21 - 18 = 3

ar[70] = 21 - 18 = 3

ar[71] = 21 - 18 = 3

ar[72] = 21 - 18 = 3

ar[73] = 21 - 18 = 3

ar[74] = 21 - 18 = 3

ar[75] = 21 - 18 = 3

ar[76] = 21 - 18 = 3

ar[77] = 21 - 18 = 3

ar[78] = 21 - 18 = 3

ar[79] = 21 - 18 = 3

ar[80] = 21 - 18 = 3

ar[81] = 21 - 18 = 3

ar[82] = 21 - 18 = 3

ar[83] = 21 - 18 = 3

ar[84] = 21 - 18 = 3

ar[85] = 21 - 18 = 3

ar[86] = 21 - 18 = 3

ar[87] = 21 - 18 = 3

ar[88] = 21 - 18 = 3

ar[89] = 21 - 18 = 3

ar[90] = 21 - 18 = 3

ar[91] = 21 - 18 = 3

ar[92] = 21 - 18 = 3

ar[93] = 21 - 18 = 3

ar[94] = 21 - 18 = 3

ar[95] = 21 - 18 = 3

ar[96] = 21 - 18 = 3

ar[97] = 21 - 18 = 3

ar[98] = 21 - 18 = 3

ar[99] = 21 - 18 = 3

ar[100] = 21 - 18 = 3

ar[101] = 21 - 18 = 3

ar[102] = 21 - 18 = 3

ar[103] = 21 - 18 = 3

ar[104] = 21 - 18 = 3

ar[105] = 21 - 18 = 3

ar[106] = 21 - 18 = 3

ar[107] = 21 - 18 = 3

ar[108] = 21 - 18 = 3

ar[109] = 21 - 18 = 3

ar[110] = 21 - 18 = 3

ar[111] = 21 - 18 = 3

ar[112] = 21 - 18 = 3

ar[113] = 21 - 18 = 3

ar[114] = 21 - 18 = 3

ar[115] = 21 - 18 = 3

ar[116] = 21 - 18 = 3

ar[117] = 21 - 18 = 3

ar[118] = 21 - 18 = 3

ar[119] = 21 - 18 = 3

ar[120] = 21 - 18 = 3

ar[121] = 21 - 18 = 3

ar[122] = 21 - 18 = 3

ar[123] = 21 - 18 = 3

ar[124] = 21 - 18 = 3

ar[125] = 21 - 18 = 3

ar[126] = 21 - 18 = 3

ar[127] = 21 - 18 = 3

ar[128] = 21 - 18 = 3

ar[129] = 21 - 18 = 3

ar[130] = 21 - 18 = 3

ar[131] = 21 - 18 = 3

ar[132] = 21 - 18 = 3

ar[133] = 21 - 18 = 3

ar[134] = 21 - 18 = 3

ar[135] = 21 - 18 = 3

ar[136] = 21 - 18 = 3

ar[137] = 21 - 18 = 3

ar[138] = 21 - 18 = 3

ar[139] = 21 - 18 = 3

ar[140] = 21 - 18 = 3

ar[141] = 21 - 18 = 3

ar[142] = 21 - 18 = 3

ar[143] = 21 - 18 = 3

ar[144] = 21 - 18 = 3

ar[145] = 21 - 18 = 3

ar[146] = 21 - 18 = 3

ar[147] = 21 - 18 = 3

ar[148] = 21 - 18 = 3

ar[149] = 21 - 18 = 3

ar[150] = 21 - 18 = 3

ar[151] = 21 - 18 = 3

ar[152] = 21 - 18 = 3

ar[153] = 21 - 18 = 3

ar[154] = 21 - 18 = 3

ar[155] = 21 - 18 = 3

ar[156] = 21 - 18 = 3

ar[157] = 21 - 18 = 3

ar[158] = 21 - 18 = 3

ar[159] = 21 - 18 = 3

ar[160] = 21 - 18 = 3

ar[161] = 21 - 18 = 3

ar[162] = 21 - 18 = 3

ar[163] = 21 - 18 = 3

ar[164] = 21 - 18 = 3

ar[165] = 21 - 18 = 3

ar[166] = 21 - 18 = 3

ar[167] = 21 - 18 = 3

ar[168] = 21 - 18 = 3

ar[169] = 21 - 18 = 3

ar[170] = 21 - 18 = 3

ar[171] = 21 - 18 = 3

ar[172] = 21 - 18 = 3

ar[173] = 21 - 18 = 3

ar[174] = 21 - 18 = 3

ar[175] = 21 - 18 = 3

ar[176] = 21 - 18 = 3

ar[177] = 21 - 18 = 3

ar[178] = 21 - 18 = 3

ar[179] = 21 - 18 = 3

ar[180] = 21 - 18 = 3

ar[181] = 21 - 18 = 3

ar[182] = 21 - 18 = 3

ar[183] = 21 - 18 = 3

ar[184] = 21 - 18 = 3

ar[185] = 21 - 18 = 3

ar[186] = 21 - 18 = 3

ar[187] = 21 - 18 = 3

ar[188] = 21 - 18 = 3

ar[189] = 21 - 18 = 3

ar[190] = 21 - 18 = 3

ar[191] = 21 - 18 = 3

ar[192] = 21 - 18 = 3

ar[193] = 21 - 18 = 3

ar[194] = 21 - 18 = 3

ar[195] = 21 - 18 = 3

ar[196] = 21 - 18 = 3

ar[197] = 21 - 18 = 3

ar[198] = 21 - 18 = 3

ar[199] = 21 - 18 = 3

ar[200] = 21 - 18 = 3

ar[201] = 21 - 18 = 3

ar[202] = 21 - 18 = 3

ar[203] = 21 - 18 = 3

```
#include <bits/stdc++.h>
using namespace std;
main()
```

```
{ int ar[10];
```

```
for (int k=0; k<10; k++)
```

```
{ ar[k] = k * 2 + 3;
```

```
if (k%2==0 // k%3==0)
```

```
ar[k] = 2 * k;
```

```
}
```

0	1	2	3	4	5	6	7	8	9
3	5	3	3	3	13	3	17	3	3

```
for (int k=0; k<10; k++)
```

```
cout << ar[k] << " ";
```

```
}
```

Initializing an Array → size accommodate the elt
→ we can omit size i.e [] ✓

- fill the array with elements (automatically)

NOTE-We can initialize an array as:-

Data Type Array Name [size] = { list of comma separated elts. }

Eg, initialize ages of 5 students

```
int age[5] = { 20, 19, 21, 23, 18 }; [5] x [10] ✓
```

initializing names of 5 students

```
string names[5] = { "aa", "cc", "bb", "ee", "dd" };
```

```
ar[0]=3
```

```
ar[0]=3-0=3
```

```
ar[1]=5
```

```
ar[2]=7
```

```
ar[3]=7-4=3
```

```
ar[3]=9
```

```
ar[3]=9-6=3
```

```
ar[4]=11
```

```
ar[4]=11-8=3
```

```
ar[5]=13
```

```
ar[6]=15
```

```
ar[6]=15-12=3
```

```
ar[7]=17
```

```
ar[8]=19
```

```
ar[8]=19-16=3
```

```
ar[9]=18+3=21
```

```
ar[9]=21-18=3
```



```
#include <bits/stdc++.h>
using namespace std;
main()
```

```
{ int age[] = {20, 19, 21, 23, 18};
// To display contents of the array;
```

```
for (int i=0; i<5; i++)
{ cout << age[i] << " ";
}
```

1. Write a c++ program that do the following;

a = add even elements of the Array c) print elements found at even index
b = print odd elements of the Array

```
int volts[] = {25, 4, 9, 27, 40, 26, 8, 5};
```

01. #include <bits/stdc++.h>

```
using namespace std;
```

```
main()
```

```
{ int volt[] = {25, 4, 9, 27, 40, 26, 8, 5}, sum=0;
```

```
for (int i=0; i<8; i++)
```

```
{ if (volts[i] % 2 == 0)
```

$i \% 2 == 0 \rightarrow$ "index" even

```
sum += volts[i];
```

```
}
cout << "sum of even elements = " << sum;
```

```
}
```

c) main()

```
{ int volt[] = {25, 4, 9, 27, 40, 26, 8, 5};
```

```
for (int i=0; i<8; i++)
```

```
{ if (volts[i] % 2 == 0)
```

```
cout << volts[i];
```

```
}
```

b) main()

```
{ int volts[] = {25, 4, 9, 27, 40, 26, 8, 5};
```

```
for (int i=0; i<8; i++)
```

```
{ if (volts[i] % 2 != 0)
```

```
{ cout << volts[i];
}
```

```
}
```

MY COLLEAGUE

Two Dimensional Array (Table) - size must be given [x]

Declaration & initialization of the array is the same as single dimension array, except its dimension

- During initialization size is Not optional unlike single dimension array.

- To access elements of the array we use **NESTED LOOPS**.

Two Dimensional Array Declaration

Data Type Array Name [rows] [columns];

				column	0 → N-1
			row		
eg	0	1	2	3	
0	20	5	10	6	
1	40	7	20	4	(20)
2	10	9	30	0	[1][2]

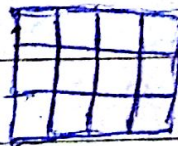
Ex Write a C++ program that inputs the data into a two dimensional array, ^(3x4) named temperature, Then your program should prints the element in a tabular format { 20, 5, 10, 6, 40, 7, 20, 4, 10, 9, 30, 0 }

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
main()
```

```
{ int temperature[3][4];
```



```
for (int r=0; r<3; r++)
```

```
{ for (int c=0; c<4; c++)
```

```
cin >> temperature[r][c];
```

```
}
```


Print the pit

Chapter 3(7)

```
for (int r=0; r<3; r++)
{
    for (int c=0; c<4; c++)
    {
        cout << temperature[r][c] << " ";
        cout << endl;
    }
}
```

out put

20	5	10	6
40	7	30	4
10	9	30	0

eg, write a program that adds the 3x3 matrix.

$$A = \begin{bmatrix} 1 & 4 & 3 \\ 2 & 5 & 2 \\ 3 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Solⁿ

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
main()
```

```
{
    int A[3][3] = {1, 4, 3, 2, 5, 2, 3, 0, 1};
    int B[3][3] = {1, 0, 1, 0, 1, 0, 1, 0, 1};
    int C[3][3];
```

```
// Lets fill matrix C
```

```
for (int r=0; r<3; r++)
{
    for (int c=0; c<3; c++)
    {
        C[r][c] = A[r][c] + B[r][c];
    }
}
```

```
// To Display contents of matrix C
```

```
for (int r=0; r<3; r++)
{
    cout << endl;
    for (int c=0; c<3; c++)
    {
        cout << C[r][c] << " ";
    }
} // End of main()
```

MY COLLEAGUE

Op

Passing Array As an Argument

- Like any variable an array can also be passed on to the called fun as an argument.

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
void show(int m[], n)
```

```
{ for(int i=0; i<n; i++)
```

```
    cout << m[i] << " ";
```

```
}
```

```
main()
```

```
{ int ar[] = {2, 7, 5, 9, 13, 40, 8, 1, 25, 10};
```

```
    show(ar, 10); // fun calling
```

```
}
```

(ar, 5)

Operation on Array

Sorting — { selection
bubble

Searching — { linear
binary — the most efficient

mid pt

if x > mid pt
search to the right

Sorting — ordering elements

→ Selection (Ascending order A-Z)

for un-ordered collection

- Find the smallest element & put it at the beginning
- From the remaining unordered collection find the smallest & put it at the 2nd place etc - -

eg. [10, 9, 20, 70, 6, 25, 4, 35, 1, 5]

Eg: [10, 9, 20, 7, 6]

= 9, 10, 20, 7, 6

= 9, 10, 7, 20, 6

= 9, 10, 7, 10, 20, 6

= 7, 9, 10, 20, 6

= 7, 9, 10, 6, 20

7, 9, 10, 10, 20

7, 6, 9, 10, 20

6, 7, 9, 10, 20

9 10 20 7 6 25 4 35 1 5

6 7 9 10 20 25 4 35 1 5

25 7 9 10 20 4 35 1 5

4 7 9 10 20 35 1 5

35 7 9 10 20 1 5

35 1 7 9 10 20

1 5 7 9 10 20

```
#include <bits/stdc++.h>
using namespace std;
```

```
int ar[] = {10, 9, 20, 7, 6, 25, 4, 35, 1, 5};
int temp = 0;
// Sort the elements
```

```
for (int i = 0; i < 10; i++)
```

```
{ for (int j = i + 1; j < 10; j++)
```

```
{ if (ar[j] < ar[i])
```

```
temp = ar[j];
```

```
ar[j] = ar[i];
```

```
ar[i] = temp; ar[j] = ar[j] + put
```

```
cout << ar[i] << " ";
```

```
}
```

or

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
main()
```

```
{ int ar[] = {10, 9, 20, 7, 6, 25, 4, 35, 1, 5};
```

Ascending sort (ar, ar + 10);

```
for (int i = 0; i < 10; i++)
```

```
cout << ar[i] << " ";
```

```
}
```

Bubble

for unordered collection

→ check the adjacent elements

if they are out of order it swap them

Repeat the above procedure until all the elements become ordered

Descending

```
sort(ar, ar + 10);
```

```
for (int i = 9; i >= 0; i--)
```

```
cout << ar[i] << " ";
```

MY COLLEAGUE


```
#include <bits/stdc++.h>
using namespace std;
```

```
main()
{
    int ar[10] = {10, 9, 20, 30, 6, 25, 4, 35, 1, 5};
    // sort the element using bubble sort
    for (int i = 0; i < 10; i++)
    {
        for (int j = i + 1; j < 10; j++)
        {
            if (ar[j] < ar[i])
                swap(ar[i], ar[j]);
        }
    }
    // Display sorted elements
    for (int i = 0; i < 10; i++)
        cout << ar[i] << " ";
}
```

~~EW~~ ~~using~~ ~~int & m;~~ ~~EW~~

4	9	8
7	9	5
9	10	7
5	9	3

#1. write a c++ program that prints the above table

#2. that ~~works~~ ^{adds} elements of the table

#3. that finds & prints the maximum & minimum values of the table.

Soln

1. main()

```
{ int a[4][3] = {4, 9, 8, 7, 9, 5, 9, 10, 7, 5, 9, 3};
```

```
for (int r = 0; r < 4; r++)
```

```
{ for (int c = 0; c < 3; c++)
```

```
cout << a[r][c] << " ";
```

```
cout << endl;
}
```

```
{ for (int c = 0; c < 3; c++)
```

```
{ sum += a[r][c];
```

```
cout << a[r][c] << " ";
```

```
cout << endl;
```

```
cout << sum;
```

2

main()

```
{ int a[4][3] = {4, 9, 8, 7, 9, 5, 9, 10, 7, 5, 9, 3}; sum = 0;
```

```
for (int r = 0; r < 4; r++)
```

MY COLLEAGUE

out 85


```

min
main()
{ int a[] = {4, 9, 8, 7, 9, 5, 9, 10, 7, 5, 9, 3}, min = a[0];
  for (int i = 0; i < 12; i++)
  { if (a[i] < min)
    { min = a[i];
    }
  }
  cout << min << endl;
}

```

```

max
main()
{ int a[] = {4, 9, 8, 7, 9, 5, 9, 10, 7, 5, 9, 3}, max = a[0];
  for (int i = 0; i < 12; i++)
  { if (a[i] > max)
    { max = a[i];
    }
  }
  cout << max << endl;
}

```


asterisk (*)

write a program that has a declaration in `main()` to store the following numbers in an array named `temps`;
6.5, 7.2, 7.5, 8.3, 9.4, 9.6, 9.8 and 10 there should be a `fun` call to `show()` that accepts the `temps` array as a parameter named `temps` and displays numbers in the array.

```
#include <iostream>
using namespace std;
main()
void show (float temp[], int b)
{
    for (int i=0; i<b; i++)
    {
        cout << temp[i] << " ";
    }
}
main()
```

```
{
    float ar[] = {6.5, 7.2, 7.5, 8.3, 8.6, 9.4, 9.6, 9.8, 10};
    show(ar, 8);
}
```

write a c++ program that add odd elts

`vals[] = {10, 20, 5, 30, 25, 40, 50, 35}`

```
main()
```

```
{
    int vals[] = {10, 20, 5, 30, 25, 40, 50, 35}, sum=0;
    for (int i=0; i<8; i++) or for (int i=1; i<8; i+=2)
    {
        if (i%2 != 0)
        {
            sum += vals[i];
        }
    }
    cout << "sum = " << sum;
}
```

OUT PUT
125

Pointers

A kind of Variable, which stores the Address of another Variables.

Like any variables pointers have to be declared prior to using them in the Program.

Declaration

Data Type * Pointer Name;

Eg. int * p;

- NOTE pointers doesn't have data type on their own.

→ the data type is refers to the data type of the variable they point to.

(*) pointers

- Use two Operator :- $\&$ (address Operator) and * (Dereference Operator)

- $\&$ (address Operator)

- to access the address of the variable to which the pointer points to.

* (Dereference Operator)

- to access the value of the variable, to which the pointer points to.

Eg. #include <iostream>
using namespace std;
main()

{ int x = 25, y = 10;

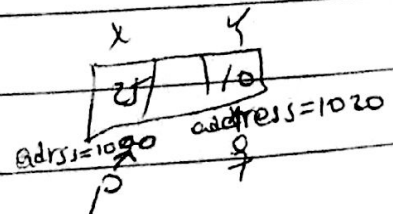
float * p, * q; ~~int * p, * q;~~

p = &x;

q = &y;

cout << p << " " << *p << endl;

cout << q << " " << *q << endl;



Output	
1000	25
1020	10

MY COLLEAGUE

Eg, #include <bits/stdc++.h>
using namespace std;
main()

{ int x=25, y=10;

int *p, *q;

p = &x;

q = &y;

cout << p << " " << *p << endl;

cout << q << " " << *q << endl;

p = q; // p points to q

cout << *p << endl << q << endl;

}

out put

10

1020

~~1020~~

q = p

#include - -

using - -

main()

{ int a=7, b=5, c=9;

*int q, *p;

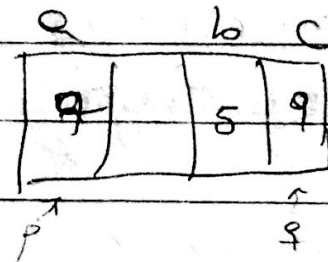
q = &c;

p = &a;

cout << a << " " << *q << endl;

cout << *q << " " << &b << endl;

}



out put

7	9
9	address of b

Pointer Arithmetics

Eg.

#inc
usr

main()

{ int x=7, y=10;

int *p, *q;

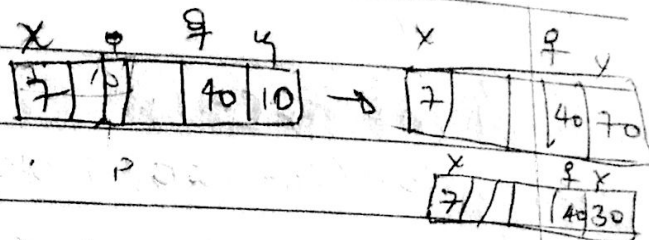
p = &y;

*p * = x; $\rightarrow$ *p = *p * x = 70

q = 40

*p = q; // *p = *p - q;

cout << x << " " << y;



out put

7 30

Eg2, #inc
usr

main()

{ int x=25, y=10;

int *p, *q;

p = &x;

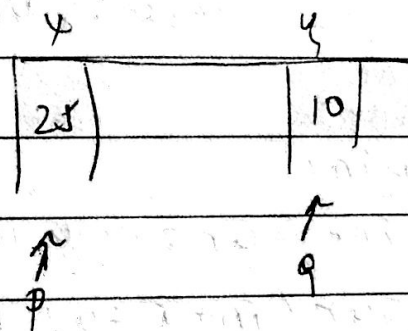
q = &y;

*p * = 3; $\rightarrow$ *p = *p * 3
25 * 3 = 75

q = p;

*q / = 8;

cout << x << " " << y << endl;



out put

p = q 75 1

q = p 9 10

Static location

Array and pointers

Ex: #inc
using
main()

```
} int ar[5];
```

To fill the array with element

```
for (int i=0; i<10; i++)  
{ cout<<"Enter element";  
  cin>>ar[i];  
}
```

To display content of the array

```
for (int i=0; i<10; i++)  
{ cout<<ar[i]<<" ";  
}
```

Verses (Vs)

Dynamic location

#inc
using
main()

```
{ int *ar = new int[5];
```

```
for (int i=0; i<10; i++)  
{ cout<<"Enter element";  
  cin>>* (ar+i);  
}
```

```
for (int i=0; i<10; i++)  
  cout<<* (ar+i)<<" ";  
  ↳ ar[i]
```


Passing Array as an argument using Pointers

eg. #inc
using
main()

```

void Show(int *vals, int x)
{
    for (int i = 0; i < x; i++)
    {
        cout << *(vals + i) << " ";
    }
    main()
}
int x[] = {19, 10, 7, 50, 30, 40, 35, 90};
Show(x, 8); or Show(&x, 8);

```

4. write a declaration to store the 7 values in an array, corresponding rates: 12.9, 18.6, 11.4, 13.7, 9.5, 15.2 & 17.6. include declaration in a program that display the values in array by using pointer notation.

main()

```

{
    float rates[] = {12.9, 18.6, 11.4, 13.7, 9.5, 15.2, 17.6};
    for (int i = 0; i < 7; i++)
        cout << *(rates + i) << " ";
}

```

find sum at even index

```

{
    float rates[] = {12.9, 18.6, 11.4, 13.7, 9.5, 15.2, 17.6};
    sum = 0;
    for (int i = 0; i < 7; i++)
    {
        if (i % 2 == 0)
            sum += rates[i];
    }
    cout << sum;
}

```


3, write a program that stores the 77 nos in the array named miles: 15, 22, 16, 18, 27, 23, 20. Have your program copy the data stored in miles to another array named dist. & then your program should use pointer notation when copying & displaying the array elts

```
main()
{
    int miles[] = {15, 22, ...};
    int dist[7];
    for (int i = 0; i < 7; i++)
    {
        *(dist + i) = *(miles + i);
        cout << *(dist + i) << " ";
    }
}
```

MULTIPLYING A MATRIX

```
main()
{
    → int a[2][3] = {1, 2, 3, 4, 5, 6};
    int b[3][2] = {5, 8, 6, 9, 7, 0};
    int c[2][2], sm;
    for (int r = 0; r < 2; r++)
    {
        for (int k = 0; k < 2; k++)
        {
            sm = 0;
            for (int c = 0; c < 3; c++)
            {
                sm += a[r][c] * b[c][k];
                cout << sm << " ";
            }
            cout << endl;
        }
    }
}
```


3. Write a program that stores the 77 nos in the array named miles: 15, 22, 16, 18, 27, 23, 20. Have your program copy the data stored in miles to another array named dist. & then your program should use pointer notation when copying & displaying the array elements.

```
main()
```

```
{ int miles[] = { 15, 22, ... };
```

```
int dist[7];
```

```
for (int i = 0; i < 7; i++)
```

```
{ * (dist + i) = * (miles + i);
```

```
cout << * (dist + i) << " ";
```

MULTIPLYING A MATRIX

```
main()
```

```
{
```

```
→ int a[2][3] = { 1, 2, 3, 4, 5 };
```

```
int b[3][2] = { 5, 8, 6, 9, 7, 0 };
```

```
int c[2][2], sm;
```

```
for (int r = 0; r < 2; r++)
```

```
{ for (int k = 0; k < 2; k++)
```

```
{ sm = 0;
```

```
for (int c = 0; c < 3; c++)
```

```
sm += a[r][c] * b[c][k];
```

```
cout << sm << " ";
```

```
}
```

```
cout << endl;
```

```
}
```

```
}
```


Find sum of elements at odd index

size[] = {10.5, 80.6, 40.3, 0, 13.7, 9.3, 8.5, 4.2}

sum = 0;

for (int i = 0; i < 8; i++)

{
if (i % 2 != 0)

sum += size[i];

}

cout << sum;

write out put the 77

A. int a[7];

a[0] = 9;

for (int k = 1; k < 7; k++)

{

a[k] = k % 3 + 5;

a[k-1] = a[k] - 3;

}

for (int k = 0; k < 7; k++)

cout << a[k] << " ";

out put

3 4 2 3 4 2 5

B. int x = 30, y = 50;

int *p, *q;

q = &x;

p = &y;

cout << *q / 3 << endl;

cout << *p << " " << *q + *p << endl;

out put

10

50

60

MY COLLEAGUE

File

- Storing the program execution result permanently for future retrieval & modification

Steps

1. Create Constructor (creating object)
2. Open the file if exist / create new
3. Read from / write on to a file
4. Close the file

Modes (in step 2)

Writing

Reading

ios::out

ios::in

ios::app

* ios::out with ^{new} file by deleting previous

- write on a file

- if the file has pre-existed contents, ios::out will overwrite the previous content with new ones

to ios::app → don't delete previous both the new & old co-exists

- write on a file

- if the file has pre-existed contents, it will appends the new entries at the end of the previous contents

Reading file

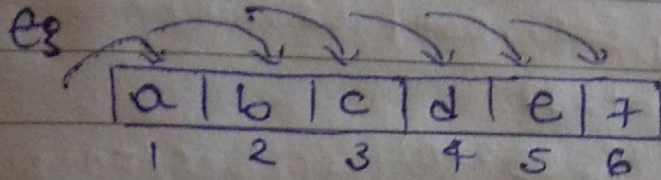
FileObject * eof() → reads contents of the file until the file reach to its final point

$\uparrow$ stream - both
 $\uparrow$ ifstream - read
 $\uparrow$ ofstream - write

Accessing file

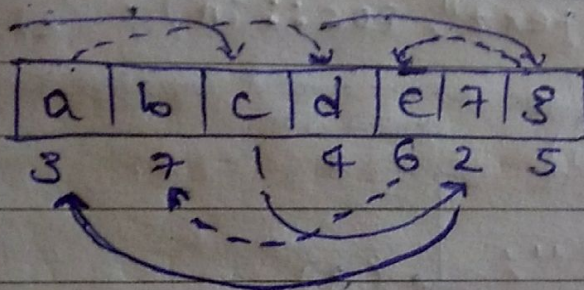
1. Sequential file accessing

- access the file from start to end sequentially



2. Random file access

- Accessing file contents randomly



* Use pointers

→ seekg() or seekp()

- tellg() or tellp()

Ex 1. write a c++ program that accepts file scores from a user, & write them on a text file called "sample text". Then, your program should retrieve the content of the file & displays it on the console

```
#include <bits/stdc++.h> // #include <iostream>
                          // #include <fstream>
using namespace std;
main()
```

```
{ fstream score;
```

```
  fstream f; // step (1)
```

```
  f.open("sample text", ios::out); // step (2)
```

```
  for (int i=1; i<=5; i++)
```

```
    fcout << "Enter scores";
```

```
    cin >> scores;
```

```
    f << score << " "; // write to a file
```

```
  f.close();
```

Step 3

Step 4

MY COLLEAGUE

let our input = 10, 25, 30.5, 40, 17.3

To read content of file

```
#include <iostream>
```

```
using namespace std;
```

```
main()
```

```
{
```

```
float score;
```

```
ifstream f; (1)
```

```
f.open("sample.txt", ios::in); (2)
```

```
f >> score;
```

```
while (!f.eof())
```

```
{ cout << score << " ";
```

```
f >> score;
```

```
}
```

```
f.close(); (4)
```

```
}
```

Ex 2. write a c++ program that writes multiple of 3
b/n 10 & 30 on a file called "multiple.txt".

Your program should read the content of the file &
prints the result.

```
main()
```

```
{ int i;
```

```
ofstream k;
```

```
k.open("multiple.txt", ios::out | ios::trunc);
```



```
for (int i=12; i<=30; i+=3) / write multiple of 3
    k << i << " ";
    k >> i;
```

```
while (!k.eof()) / reading content of
    { cout << i << " "; the file
      k >> i;
    }
```

```
k.close();
```

```
}
```

or

```
main()
```

```
{ int i;
```

```
  ofstream k;
```

```
  ofstream d;
```

```
  d.open("multiple.text", ios::out);
```

```
  for (i=12; i<=30; i+=3)
```

```
    d << i << " ";
```

```
  d.close();
```

```
  k.open("multiple.text", ios::in);
```

```
  k >> i;
```

```
  while (!k.eof())
```

```
    { cout << i << " ";
```

```
      k >> i;
```

```
    }
```

```
  k.close();
```

```
}
```

Q3. Write a C++ program that write even # b/w 20 & 50 on a file called "even.text", excluding 26 & 38. Then reads the content of that file


```

main()
{
    int i;
    ofstream k;
    k.open("even.txt", ios::out | ios::trunc);
    for (i = 20; i <= 50; i += 10)
    {
        if (i % 26 == 0 || i % 38 == 0)
        {
            continue;
            k << i << " ";
        }
        k << i << " ";
    }
    while (!k.eof())
    {
        cout << i << " ";
        k >> i;
    }
    k.close();
}

```

NB

```

ofstream k;
k.open("even.txt");

```

same → ofstream k ("even.txt");

Ex. A write a c++ program that writes the 77 table on a file called "table.txt".

B. read contents of the file, "table.txt".

table →	mid	final	Total
	2	7	9
	4	5	9
	0	7	7
	6	10	16


```

#include <bits/stdc++.h>
using namespace std;
main()
{
    int md, n, tot;
    ofstream f1("table.txt", ios::app);
    for (int k = 1; k <= 5; k++)
    {
        cin >> md >> n;
        tot = md + n;
        f1 << md << "\t" << n << "\t" << tot << endl;
    }
    f1.close();
}

```

(B)

```

main()
{
    ofstream f1("table.txt", ios::in);
    f1 >> md >> n >> tot;
    cout << "Mid\tFinal\tTotal" << endl;
    while (!f1.eof())
    {
        cout << md << "\t" << n << "\t" << tot << endl;
        f1 >> md >> n >> tot;
    }
    f1.close();
}

```

Ex 4. write a program that accept name, age & sex of 5 students & write them on file "student.txt".
 B. read content on file, "student.txt"

Character M
 string male

MY COLLEAGUE


```

main()
{
    string n;
    int a;
    char s;
    ostream k;
    k.open("student.txt", ios::app);
    for (int i=1; i<=5; i++)
    {
        cout << "Enter name, age, & sex of a student";
        getline(cin, n, '.');
        cin >> a >> s;
        k << n << "\t" << a << "\t" << s << endl;
    }
    k.close();
}

```

```

B1 istream d;
d.open("student.txt", ios::in);
d >> n >> a >> s;
cout << "Name\tAge\tGender" << endl;
while (!d.eof())
{
    cout << n << "\t" << a << "\t" << s << endl;
    d >> n >> a >> s;
}
d.close();

```

Ex Fruit.txt

Fruits	Quantity	Price
Apple	20	100
Banana	40	13
Papaya	10	22
Avocado	8	11

Passing file object as an argument to a fun
Eg. odd #s b/w 20 & 40 (A. write them on a file
"odds.txt")

B. reads contents of "odds.txt";

Solⁿ A. #include <bits/stdc++.h>
using namespace std;

void writeTofile(fstream &fs)

{ fs.open("odds.txt", ios::out);

for(int k=20; k<=40; k+=2)

{

fs << k << " ";

fs.close();

}

main()

{ fstream f;

Do reading

writetoFile(f);

}

Sort

sort(mA, mA+10)

reverse(mA, mA+10);


```
#include --
```

```
using --
```

```
main()
```

```
{ float score;
```

```
fstream f;
```

```
f.open("sample.txt", ios::out | ios::trunc);
```

```
for(int i=1; i<=5; i++)
```

```
{ cin >> score;
```

```
f << score << " ";
```

```
f.close(); system("cls");
```

```
f.open("sample.txt", ios::out | ios::trunc);
```

```
while(!f.eof())
```

```
{ f >> score;
```

```
cout << score << " ";
```

```
}
```

```
f.close();
```

```
}
```

write inputs
or file

retrieve &
prints
content or
file


```

int n, a[10], i=0;
cin >> n;

```

```

while n>0
a[i] = n%2;
n = n/2;
i++;
}

```

```

reverse(a, a+i);

```

```

for(int k=0; k<i; k++)
cout << a[k];

```

Output

1010

$i=0$
 $n=10$ $a[0] = 10\%2 = 0$
 $n = 10/2 = 5$

$i=1$ $a[1] = 5\%2 = 1$
 $n = 5/2 = 2$

$i=2$

$i=2$ $a[2] = 2\%2 = 0$
 $n = 2/2 = 1$

$i=3$ $i=3$

$a[3] = 1\%2 = 1$

0101

reverse = 1010

Serializes for only
 $n>0$

(B) int a=99, b=77, c=49;

```

int *p, *q;

```

```

p = &c;

```

```

q = &a;

```

```

cout << a << " " << b << " " << c << endl;

```

```

cout << p << endl << q;

```

```

* q += 100;

```

```

*p = *q;

```

```

cout << a << endl << b << endl;

```

```

cout << c << endl;

```

Output

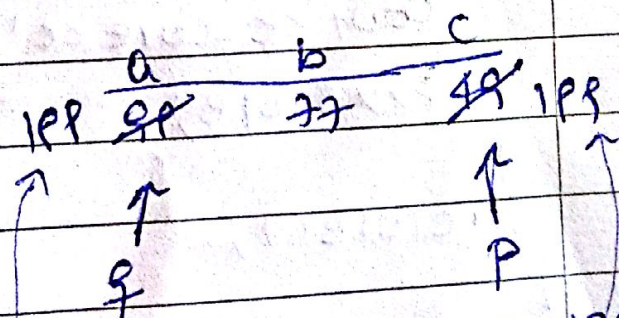
99 77 49

Address of c

Address of a 199

77

199



$*q = *p + 100 = 99 + 100 = 199$
 $*p = *q$ ($*p = 199$)

HW ↓

Ex 1# multiple.txt

↓
multiple of 5 bin 50 & 70

Ex 2 write a c++ program that writes the 77 values
on a file "quiz.txt"

[10, 43, 9, 70, 95, 60, 5]

B: Display contents of "quiz.txt" [only those
divisible by 5]

main Random read

```
{ float score;
```

```
  long pos;
```

```
  fstream f;
```

```
  f.open("Sample.txt", ios::in);
```

```
  f.seekg(0L, ios::beg);
```

```
  f >> score;
```

```
  while (!f.eof())
```

```
  { pos = f.tellg();
```

```
    f.seekg(pos);
```

```
    cout << score << " ";
```

95 60 5

```
    f >> score;
```

```
  }
```

```
  f.close();
```

```
}
```

The quick brown fox jumps over the lazy dog.

palindrome = 104397095605

main()

{ float score;

010812 10 43 9 70 95 605

long pos;

88 99 9 70 95 605

~~fstream~~ f;

f.open("sample.txt", ios::out | ios::in);

f.seek(0L, ios::beg);

for (int k=1; k<=2; k++)

{ pos = f.tellg();

f.seek(pos);

cin >> score;

f << score << " ";

}

f.close();

}

main()

{ ~~string s;~~ char s[10];

long pos;

fstream f;

f.open("sample.txt", ios::in);

f >> s;

int l;

while (!f.eof())

{ l = strlen(s);

if (l >= 5)

cout << s << " ";

k = f.get();

}

f.close();

}

Out put

Quick brown jumps

MY COLLEAGUE

HW2 Write a C++ program that reads a string (word) & determines whether the word is palindrome or Not

Answer to HW1

```
1. #include <bits/stdc++.h>
using namespace std;
main()
{
    int i;
    fstream f;
    f.open("Multiple.txt", ios::out);
    for (i=50; i<=70; i+=5)
    {
        f<<i<<" ";
    }
    f.close();
    f.open("multiple.txt", ios::in);
    f>>i;
    while (!f.eof())
    {
        cout<<i<<" ";
        f>>i;
    }
    f.close();
}
```

Output

50 55 60 65 70

Structure → creating our own data type

↳ belongs to diff data types
↳ global

- A data structure used to store heterogeneous yet ~~int~~ interrelated elements

Declaration

- Like any variable it has to be declared before we put it into operation.

```
struct    struct name
```

```
{
```

```
    member variables
```

```
};
```

Eg. Declaring a structure student that has number, variables, ID, name, dept.

```
⇒ struct student
```

```
{ int id;
```

```
    string name, dept
```

```
};
```

Accessing member variables

- To access member variables we use a structure Object

Eg. write a program that access member variables of a structure student

```
#include
```

```
using
```



```
#include
```

```
using
```

```
struct student
```

```
{  
    int id = 101;
```

```
    string name = "AAA"; dept = "CSE";
```

```
}
```

```
main()
```

```
{  
    ad. type student s; // structure object
```

```
// we can access every number/variable
```

```
// via s
```

```
    cout << s.id << endl;
```

```
    cout << s.name << endl;
```

```
    cout << s.dept << endl;
```

```
}
```

```
or
```

```
s: s
```

```
{
```

```
    int id = 101;
```

```
    string name = --
```

```
} s;
```

```
main()
```

```
{
```

```
    omit student s; b/c
```

```
// we can ---
```


Ex_A: write a c++ program that declare a structure student with member variables (id, name, mid, fin). Your program should initialize values to member variables then prints the number variables value.

Ex_B: Modify the above program so that it accepts the values of member variables from the user.

Soln

```
A. #include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct student
```

```
{ int id;
```

```
float mid, fin;
```

```
string name;
```

```
};
```

```
main()
```

```
{ // Initializing member vars
```

```
s = { 101, 29.7, 45.3, "AAA" }; or
```

```
// Displaying
```

```
cout << s.id << "\t" << s.name << "\t";
```

```
cout << s.mid << "\t" << s.fin << endl;
```

```
}
```

```
cout << "Enter id, name, mid & fin";  
cin >> s.id >> s.name >> s.mid >> s.fin;
```

Ex₀: write a c++ program that creates a structure fruits with member variables (fruit name, quantity & unit price) & Display number variables.

Q. write a c++ program that declare a structure car with the 7 member variables.

(plate #, color, year mtr) & try to access each number variables & display them.

MY COLLEAGUE

Ans
①

```
#include <bits/stdc++.h>
using namespace std;

struct fruit
{
    string name;
    int q;
    float p;
} f;

main()
{
    f = {"mango", 2, 12.50};
    cout << f.name << " " << f.q << " " << f.p << endl;
}
```

②

```
#include <bits/stdc++.h>
using namespace std;

struct Car
{
    int pn;
    string color;
    int ym;
} c;

main()
{
    cin >> c.pn >> c.color >> c.ym;
    cout << c.pn << " " << c.color << " " << c.ym << endl;
}
```

Enter

Array of Structure

Eg. write a c++ program that declares array of struct
a structure called student for 5 students with the following
member variables.

[id, name, score]

Soln

```
#include <bits/stdc++.h>
using namespace std;
struct student
{
    int id;
    string name;
    float score;
}
main()
{
    // Initialization of obj
    student k[5] = { {101, "AAA", 92.5}, {102, "BBB", 43.5},
                    {103, "ccc", 59}, {104, "DDD", 67.5},
                    {105, "EEE", 47} };
    // Displaying
    cout << "Id\t Name\t score" << endl;
    for (int i = 0; i < 5; i++)
        cout << k[i].id << "\t" << k[i].name << "\t" << k[i].score << endl;
    // end of main
}
```

Array k[5][3] ←

d-type obj

(2) write a c++ program that creates a structure student
for 5 students with the following member variables,

id, name, mid, fin

Then your program should accept values for member
variables & print them in a table format as;

ID	Name	mid	Final	Total
----	------	-----	-------	-------

MY COLLEAGUE


```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct Student
```

```
{ int id;
```

```
  string name;
```

```
  float md, fn;
```

```
};
```

```
main()
```

```
{ no students Student a[5];
```

```
  for(int k=0; k<5; k++)
```

```
  { cout << "Enter id, name, mid & final ";
```

```
    cin >> a[k].id >> a[k].name >> a[k].md >> a[k].fn;
```

```
  }
```

```
// Displaying
```

```
cout << "id \t Name \t Mid \t Final \t Total" << endl;
```

```
for(int k=0; k<5; k++)
```

```
{ cout << a[k].id << "\t" << a[k].name << "\t" << a[k].md << "\t"
```

```
  cout << a[k].fn << "\t" << (a[k].md + a[k].fn) << endl;
```

```
or
```

```
float total = a[k].md + a[k].fn; or total
```

```
}
```

```
} // End of main
```

Ex. do for 6 fruits

```
struct Fruit
```

```
{ string name;
```

```
  int q;
```

```
  float p;
```

```
};
```



```

main()
{
    struct a[6];
    for (int k=0; k<6; k++)
    {
        cin >> a[k].name >> a[k].g >> a[k].p;
    }

    cout << "name\tg\tp" << endl;
    for (int k=0; k<6; k++)
    {
        cout << a[k].name << "\t" << a[k].g << "\t" << a[k].p << endl;
    }
}

```

⑤ create a str called olympic that has the 77 member variables :- nation, gold, silver, bronze

Your program should accept s value for each member variables & finally displays it as

Nation G S B T

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct olympic
```

```
{
```

```
    string nation;
```

```
    int gold, silver, bronze;
```

```
};
```

```
main()
```

```
{
    olympic b[5];
```

```
    for (int k=0; k<5; k++)
```

```
    {
        cin >> b[k].nation >> b[k].gold >> b[k].silver >> b[k].bronze;
```

```
    }
    cout << "nation\tgold\tsilver\tbronze" << endl;
```

```
    for (int k=0; k<5; k++)
```

```
    {
        cout << b[k].nation << "\t" << b[k].gold << "\t" << b[k].silver << "\t" << b[k].bronze << endl;
    }
}
```

MY COLLEAGUE

NESTED STRUCTURE

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct Date
```

```
{  
    int d, y;           // int d=25, y=1992;  
    string m;           // string m="August";  
};
```

```
struct Student
```

```
{  
    int id;              // id=10;  
    string name, dept;   // string name="AAA", dept="CSE";  
    Date bd;             // Date bd;  
};
```

```
main()
```

```
{  
    Student S;  
    // Display Results  
    cout << "ID:- " << S.id << endl;  
    cout << "Name:- " << S.name << endl;  
    cout << "Dept:- " << S.dept << endl;  
    cout << "Date:- " << S.bd.m << " " << S.bd.d << " " <<  
        S.bd.y << endl;    // August 25.1992  
};
```


Ex1 Write a C++ program that declare alternative Employee with the 77 member Variables:-

id, name, position, address,

Assume that the member variable "address" has the 77 alternatives Subcity, pebele, house no & telephone

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct address
```

```
{  
    string s = "nitai silk la 7to", tell = "0900000000";
```

```
    int k = 10, hn = 140;
```

```
};
```

```
struct employee
```

```
{  
    int id = 101;
```

```
    string n = "aiaa", p = "doctor";
```

```
    address ea;
```

```
};
```

```
main()
```

```
{ employee s;
```

```
    cout << "id :- " << s.id << endl;
```

```
    cout << "name :- " << s.n << endl;
```

```
    cout << "position :- " << s.p << endl;
```

```
    cout << "address :- " << s.ea.k << " " << s.ea.s << endl;
```

```
    cout << s.ea.tell << " " << s.ea.s << endl;
```

```
}
```

Output

id :- 101

name :- aiaa

position :- doctor

Address :- 140 10 0900000000 nitai silk la 7to

MY COLLEAGUE

Ex1 Write a c++ program that declare alternative

Employee with the 77 Member Variables:-

id, name, position, address,

Assume that the member variable "address" has the 77 alternatives Subcity, kebele, house no & telephone

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct address
```

```
{  
    string s = "nitas silk 127to", tell = "092222222222";
```

```
    int k = 10, hn = 140;
```

```
};
```

```
struct employee
```

```
{  
    int id = 101;
```

```
    string n = "aaa", p = "doctor";
```

```
    address ea;
```

```
};
```

```
main()
```

```
{  
    employee s;
```

```
    cout << "id :- " << s.id << endl;
```

```
    cout << "name :- " << s.n << endl;
```

```
    cout << "position :- " << s.p << endl;
```

```
    cout << "address :- " << s.ea.hn << " " << s.ea.k << " "
```

```
    << s.ea.tell << " " << s.ea.s << endl;
```

```
}
```

Output

id :- 101

name :- aaa

position :- doctor

address :- 140 10 092222222222 nitas silk 127to

MY COLLEAGUE

Ex1 Write a C++ program that declare alternative employee with the 77 member variables:-

id, name, position, address;

Assume that the member variable "address" has the 77 alternatives Subcity, kebele, house no & telephone

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct address
```

```
{  
    string s = "nitaa sikk kotto", tell = "0900000000";
```

```
    int k = 10, hn = 140;
```

```
};
```

```
struct employee
```

```
{  
    int id = 101;
```

```
    string n = "aqa", p = "doctor";
```

```
    address ea;
```

```
};
```

```
main()
```

```
{  
    employee s;
```

```
    cout << "id :- " << s.id << endl;
```

```
    cout << "name :- " << s.n << endl;
```

```
    cout << "position :- " << s.p << endl;
```

```
    cout << "address :- " << s.ea.hn << " " << s.ea.k << " " << s.ea.s << endl;
```

```
    cout << s.ea.tell << " " << s.ea.s << endl;
```

```
}
```

Output

id :- 101

name :- aqa

position :- doctor

address :- 140 10 0900000000 nitaa sikk kotto

MY COLLEAGUE