Toppersnotes
Unleash the topper in you

बेसिक कम्प्यूटर अनुदेशक

राजस्थान कर्मचारी चयन बोर्ड, जयपुर

भाग - 5

कम्प्यूटर अध्ययन - 2

SIERRA INNOVATIONS

COMPUTER INSTRUCTOR

कम्प्यूटर अध्ययन – 2

Data Structure

- Data structure किसी कम्प्यूटर सिस्टम में डाटा को स्टोर तथा व्यवस्थित (organize) करने का एक तरीका होता है । जिससे कि हम डाटा का आसानी से इस्तेमाल कर सकें । अर्थात् डाटा को इस प्रकार स्टोर तथा organize किया जाता है कि उसको बाद में किसी भी समय आसानी access से किया जा सकें ।
- डाटा स्ट्रक्चर C, C++, Java की तरह कोई programming भाषा नहीं है बल्कि यह algorithms का एक set है जिसका प्रयोग हम programming languages मे data को structure करने के लिए करते है ।
- Data structure बहुत सारें computer science algorithms का एक मुख्य भाग है जिसके द्वारा programmers डाटा को अच्छे ढंग से handle कर सकते है। यह program या software की performance को बेहतर करने में बहुत ही अहम भूमिका निभाता है ।

Data structure दो प्रकार के होते है-

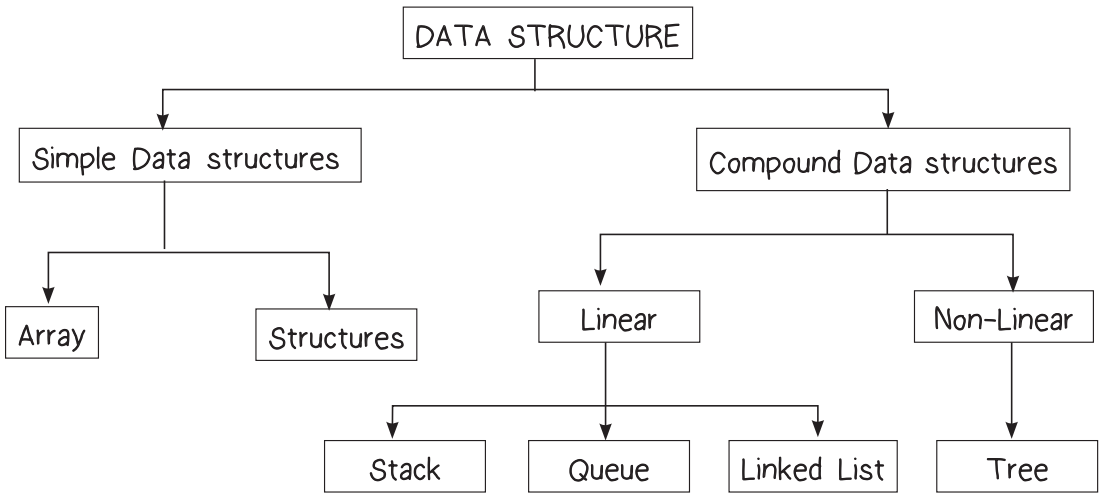
1. Primitive डाटा स्ट्रक्चर
2. Non-Primitive डाटा स्ट्रक्चर

1. Primitive डाटा स्ट्रक्चर्सै- Primitive डाटा स्ट्रक्चर वह डाटा स्ट्रक्चर होता है जिसे direct ही मशीन instructions से operate किया जा सकता है । अर्थात यह सिस्टम तथा के compiler द्वारा डिफाइन होता है ।
2. Non-Primitive डाटा स्ट्रक्चर्सै- Primitive डाटा स्ट्रक्चर वह डाटा स्ट्रक्चर होता है जिसे direct मशीन instructions से operate नही किया जा सकता है । ये डाटा स्ट्रक्चर Primitive डाटा स्ट्रक्चर से derived होते है । Non-Primitive डाटा स्ट्रक्चर दो प्रकार का होता है-

   1- Linear डाटा स्ट्रक्चर

   2- Non-linear डाटा स्ट्रक्चर

डाटा स्ट्रक्चर का वर्गीकरण-



सरल डाटा स्ट्रक्चर (Simple Data Structure)

- ये आम तौर पर प्रिमिटिव डाटा टाइप्स जैसे इंटिजर्स, रियल, करैक्टर, बूलियन से बनाया जाता है । सरल डाटा स्ट्रक्चर निम्नलिखित दो प्रकार के होते है-
  1. ऐरे (Array)
  2. स्ट्रक्चर (Structure)

यौगिक डाटा स्ट्रक्चर (Compound Data Structure)

सरल डाटा स्ट्रक्चर को विभिन्न तरीकों में संयोजित करके जटिल डाटा स्ट्रक्चर बनाये जा सकते है । ये निम्नलिखित दो प्रकार के होते है-

रेखीय डाटा स्ट्रक्चर (Linear Data Structure)

ये एकल स्तर के डाटा स्ट्रक्चर होते है । इनके तत्व एक अनुक्रम (सीक्वेंस) बनाते है इसलिए इन्हें रेखीय डाटा स्ट्रक्चर कहते है ।

ये निम्नलिखित प्रकार के होते है-

1. स्टैक (Stack)
2. क्यू लिंक लिस्ट (Queque linked list)

गैर रेखीय डाटा स्ट्रक्चर (Non-Linear Data Structure)

ये बहुतस्तरीय डाटा स्ट्रक्चर होते है । गैर रेखीय डाटा स्ट्रक्चर के उदाहरण ट्री और ग्राफ है।

- डाटा स्ट्रक्चर पर ऑपरेशन- डाटा स्ट्रक्चर पर किये जाने वाले बुनियादी ऑपरेशन इस प्रकार है-
  - इनसेशन (Insertion)- इनसेशंन का अर्थ एक डाटा स्ट्रक्चर मे एक नये डाटा तत्व को जोडना ।
  - डिलिशन (deletion)- डिलिशन का अर्थ एक डाटा स्ट्रक्चर मे एक डाटा तत्व को हटाना यदि वह मौजूद है ।
  - सर्च (Search)- एक डाटा स्ट्रक्चर में निर्दिष्ट डाटा तत्व को खोजने को सर्च कहते है ।
  - ट्रवर्सिंग (Traversing)- एक डाटा स्ट्रक्चर में मौजूद सभी डाटा तत्वों के प्रसंस्करण (प्रोसेसिंग) को ट्रेवर्सिंग कहते है ।
  - सोर्टिंग (Sorting)- डाटा स्ट्रक्चर के तत्वों को एक निर्दिष्ट क्रम में व्यवस्थित करने को सोर्टिंग कहते है ।
  - मर्जिंग (Merging)- दो एक ही प्रकार के डाटा स्ट्रक्चर के तत्वों का संयोजन कर उसी प्रकार के एक नये डाटा स्ट्रक्चर बनाने को मर्जिंग कहते है ।

Algorithm for Problem Solving

- एक एल्गोरिदम एक निश्चित पूर्वनिर्धारित कार्य को पूरा करने के लिए, इंस्ट्रक्शन या तर्क का एक परिमित सेट है, जिसे क्रम में लिखा गया है।
- एल्गोरिदम पूरा कोड या प्रोग्राम नहीं है। यह सिर्फ एक समस्या का मूल तर्क (समाधान) है, जिसे या तो एक अनौपचारिक उच्च स्तरीय विवरण के रूप में Pseudocode कोड के रूप में या फ्लोचार्ट का उपयोग करके व्यक्त किया जा सकता है।
- प्रत्येक एल्गोरिदम को निम्नलिखित गुणों को पूरा करना चाहिए

1. **Input-** There should be 0 or more inputs supplied externally to the algorithm.
2. **Output-** There should be at least 1 output obtained.
3. **Definiteness-** Every step of the algorithm should be clear and well defined.
4. **Finiteness-** The algorithm should have finite number of steps.
5. **Correctness-** Every step of the algorithm must generate a correct output.

- एक एल्गोरिदम को कुशल और तेज कहा जाता है, अगर इसे निष्पादित करने में कम टाइम लगता है और कम मेमोरी स्पेस की खपत होती है। एक एल्गोरिदम का प्रदर्शन निम्नलिखित गुणों के आधार पर मापा जाता है –
- स्पेस कॉम्पलेसिंटी
- टाइम कॉम्पलेसिटी

स्पेस जटिलता (Space Complexity)

- एल्गोरिदम के निष्पादन (Execution) के दौरान इस्तेमाल की जाने वाली आवश्यक मेमोरी या स्पेस को Space Complexity कहते हैं।
- जब Multiple User के लिए सीमित रूप से मेमोरी उपलब्ध हो तब Space Complexity आवश्यक हो जाती है।
- एक एल्गोरिदम की Space Complexity को **Big O (O(n))** Notation के द्वारा व्यक्त किया जाता है।
- आमतौर पर एक एल्गोरिदम को निम्न घटकों के लिए मेमोरी की आवश्यकता होती है
  1. इंस्ट्रक्शन स्पेस (Instruction Space½ – जब प्रोग्राम निष्पादित (Execute) होता है। तब वह जो मेमोरी या स्पेस उपयोग में लेता है। उसे इंस्ट्रक्शन स्पेस कहते हैं।
  2. डाटा स्पेस (Data Space½ – यह सभी कॉस्टेंट और वेरिएबल मानों को स्टोर करने के लिए आवश्यक स्पेस है।

समय जटिलता (Time Complexity)

- यह प्रोग्राम के पूर्ण निष्पादन (Execution) के लिए आवश्यक समय का प्रतिनिधित्व करने का एक तरीका है अर्थात् यह एल्गोरिदम के द्वारा अपनी प्रोसेस को पूरा करने में लगने वाले कुल समय की मात्रा है।
- एल्गोरिदम की टाइम जटिलता को सबसे अधिक व्यक्त करने के लिए Big O संकेतन का उपयोग किया जाता है।
- एल्गोरिदम को विकसित करने की प्रमुख श्रेणियाँ (Categories) Sort, Search, Insert, Delete, Update आदि है।

Example 1

Write an algorithm and flowchart for printing number from 1 to 20.

Algorithm -

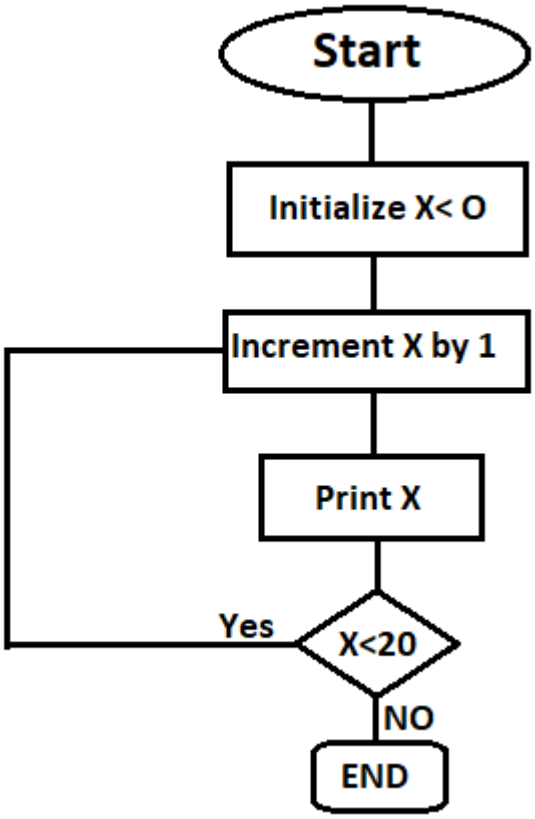
Step 1: Initialize X as 0,

Step 2: Increment X by 1,

Step 3: Print X,

Step 4: If X is less than 20 then go back to step 2.

Flowchart -



Example 2

Write an algorithm and draw the flowchart for finding the average of two numbers

Algorithm -

Step 1: input x

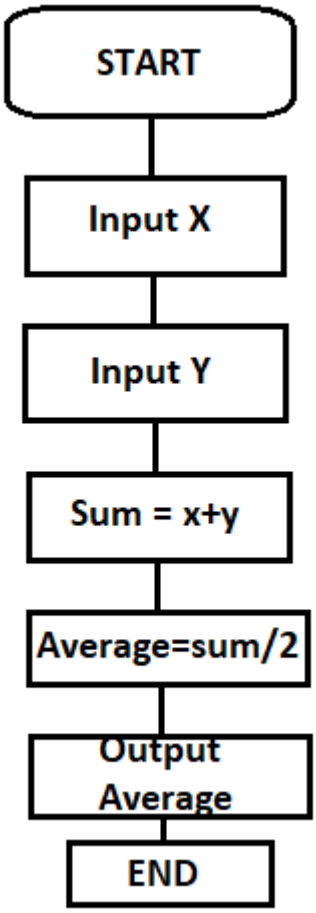
Step 2: input y

Step 3: sum = x + y

Step 4: average = sum/2

Step 5: output average

Flowchart -



Array as Data Structure

- Array एक non-primitive तथा linear डाटा स्ट्रक्चर है जो कि एकसमान (similar) डाटा items का समूह होता है, अर्थात् यह सिर्फ एक ही प्रकार के डाटा को ही स्टोर करेगा (या तो यह सिर्फ सभी integer डाटा को स्टोर करेंगा या फिर सभी floating point को)।
- Array डाटा स्ट्रक्चर का प्रयोग डाटा ऑब्जेक्ट्स के समूह को संग्रहित करने के लिए किया जाता है ।
- "Array एक static डाटा स्ट्रक्चर है अर्थात् हम केवल compile time में ही मेमोरी को allocate कर सकते है और इसे run-time में बदल नहीं सकते है ।"

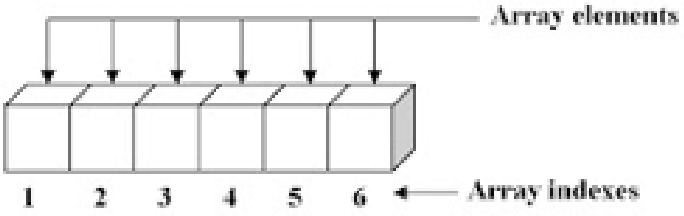
ऐरे के प्रकार (Types of array)

Array निम्नलिखित तीन प्रकार के होते है-

1. One dimensional array
2. Two dimensional array
3. Multi dimensional array

One Dimensional (1-D) Arrays

- वह arrays जिसमे सिर्फ एक subscript होती है उसे one dimensional array कहते है । इसका प्रयोग linear रूप में डाटा को स्टोर करने के लिए किया जाता है ।



One-dimensional array with six elements

एक आयामी ऐरे की घोषणा (डिक्लेरेशन)

- किसी भी अन्य वैरिएबल की तरह, ऐरे को भी उपयोग से पहले डिक्लेयर किया जाना चाहिए ताकि कम्पाइलर उनके लिए मेमोरी में स्पेस आंवटित कर सके । ऐरे को निम्न प्रकार से डिक्लेयर किया जाता है -

type variable-name [size]

उदाहरण-

int groupt[10];

float height[50];

char name[10];

टाइप ऐरे में संग्रहित होने वाले तत्वों के प्रकार को बताता है जैसे कि int, float और char एवं वैरिएबल नेम ऐरे के नाम को बताता हैजैसे कि height, group और name है साईज ऐरे में संग्रहित किये जा सकने वाले तत्वों कि अधिकतम संख्या को इंगित करता है । सी प्रोग्रामिंग भाषा, करेक्टर स्ट्रींग की करेक्टर के ऐरे के रूप में ही प्रबंध करता है ।

एकल या एक आयामी ऐरे का प्रारंभ

एक ऐरे के डिक्लेरेशन के बाद उसके तत्व प्रांरभ किये जाते है सी प्रोग्रामिंग में एक ऐरे निम्न चरणों में प्रारंभ किया जा सकता है-

- कंपाइल टाइम
- रन टाइम
- कंपाइलटाइम प्रारंभ- जब एक ऐरे के डिक्लेरेशन के साथ उसे प्रारंभ किया जाता है तो ऐरे निम्न प्रकार से प्रारंभहोगा :type array-name [size] = {list of values};

लिस्ट में मानो को कोमा से अलग किया जाता है उदहारण के लिए

int number [3] = {0, 5, 4};

ऊपर दिए गए स्टेटमेंट मे 3 आकार का एक नंबर नाम का ऐरे है और हर तत्व को वैल्यू आवंटित होगी । लिस्ट में वैल्यू की संख्या ऐरे साइज की तुलना में कम है, तो यह केवल कुछ ऐरे तत्वो को वैल्यू आवंटित करेगा । शेष तत्वों को स्वचालित रूप से शून्य आवंटित हो जायेगा ।

यदि रखें, यदि घोषित आकार की तुनला में अधिक वैल्यू है, तो एक त्रुटि का उत्पादन होगा ।

- **रन टाइम प्रारंभ-** एक ऐरे को स्पष्ट रूप से चलाने के लिए रन टाइम आंरभ किया जा सकता है । उदाहरण के लिए निम्नलिखित सी प्रोग्राम के खंड पर विचार करें ।

```
for(i=0;i<10;i++)
{
scanf("%d"; &x[i]);
}
```

एक आयामी ऐरे का प्रोग्राम-

```
/ * ऐरे में तत्वो को स्टोर करने और प्रिंट करने के लिए सरल C प्रोग्राम * /
    #include <stdio.h>
    void main()
    {
int array [5],i;
        printf("Enter 5 numbers to stores them in array \n");
```

```
for(i=0;i<5;i++)
{
    scanf("%d", &array[i]);
}
    print("Element in the array are -\n \n");
for(i=0;i<5;i++)
{
    print("Element stored at a [% d]=% d\n", i,array[i]);
}
    getch();
}
```

इनपुट (Input)- Enter 5 elements in the array- 23 45 32 25 45

आउटपुट (Output)- Elements in the array are-

Element stored at a [0]-23

Element stored at a [1]-45

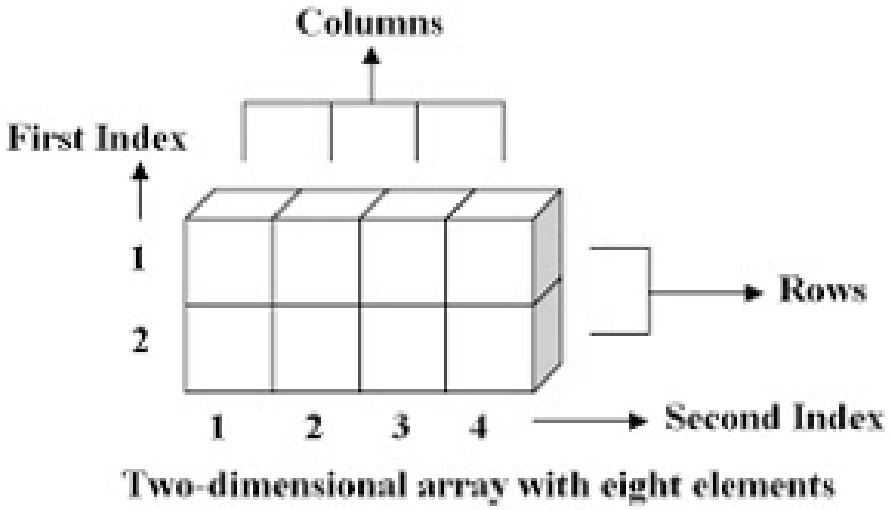
Element stored at a [2]-32

Element stored at a [3]-25

Element stored at a [4]-45

Two dimensional (2-D) Arrays

- वह array जिसमें दो subscript होती है उसे two dimensional array कहते है । two dimensional arrays को matrix तथा table भी कहते है ।



Two-dimensional array with eight elements

दो आयामी (2डी) ऐरे का प्रारंभ

एक आयामी ऐरे की तरह, 2डी ऐरे को भी दोनों प्रकार (कंपाइल टाइम व रन टाइम) से प्रारंभ किया जा सकता है

- **कंपाइल टाइम आरंभीकरण–** जब एक ऐरे के डिक्लेरेशन के साथ उसे प्रारंभ किया जाता है तो दो आयामी ऐरे निम्न प्रकार से प्रारंभहोगा :

```
int table-[2][3] = {
{0, 2, 5}
{1, 3, 0}
};
```

- रन टाइम आरंभीकरण- एक ऐरे को स्पष्ट रूप से चलाने के लिए रन टाइम आरंभ किया जा सकता है । दो आयामी ऐरे को लूप स्ट्रक्चर की मदद से आरम्भ करते हैं । दो लूप स्ट्रक्चर उपयोग में ली जाती है । जिसमें आउटर लूप पंक्ति के लिए एवं इनर लूप कॉलम के उपयोग में आती है । उदाहरण के लिए निम्नलिखित सी प्रोग्राम के खंड पर विचार करें ।

```
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
}
scanf("%d"; &ar[i] [j]);
```

```
}
}
```

2डी ऐरे का प्रोग्राम

```
/* 2-डी ऐरे का सी प्रोग्राम */
        #include <stdio.h>
        #include <stdio.h>
        void main()
    {
        int array [3] [3],i,j,count=0;
        /*Run time Initilization */
    for(i=1;i<3;i++)
            {
    for(j=1;j<3;j++)
    {
    count++;
    array[i] [j] =count;
    print("% d\t",array[i] [j]);
    }
    print("\n");
    }
        getch();
    }
    Output-
```

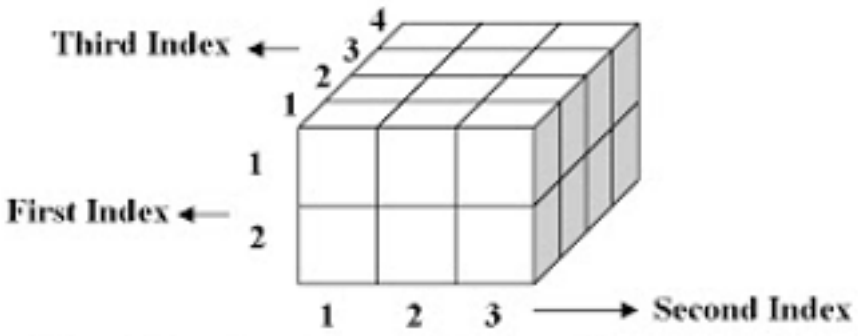
1 2 3

4 5 6

7 8 9

Multi dimensional Arrays

वह array जिसमें दो से ज्यादा subscript होती है वह multi-dimensional array कहलाता है ।



Three-dimensional array with twenty four elements

- **बहु आयामी ऐरे–** ऐरे का ऐरे एक बहुआयामी ऐरे कहलाता है । सामान्य रूप से बहुआयामी ऐरे की घोषणा निम्न प्रकार से होती है ।

  type variable-name (size1) (size2)---(sizeN);

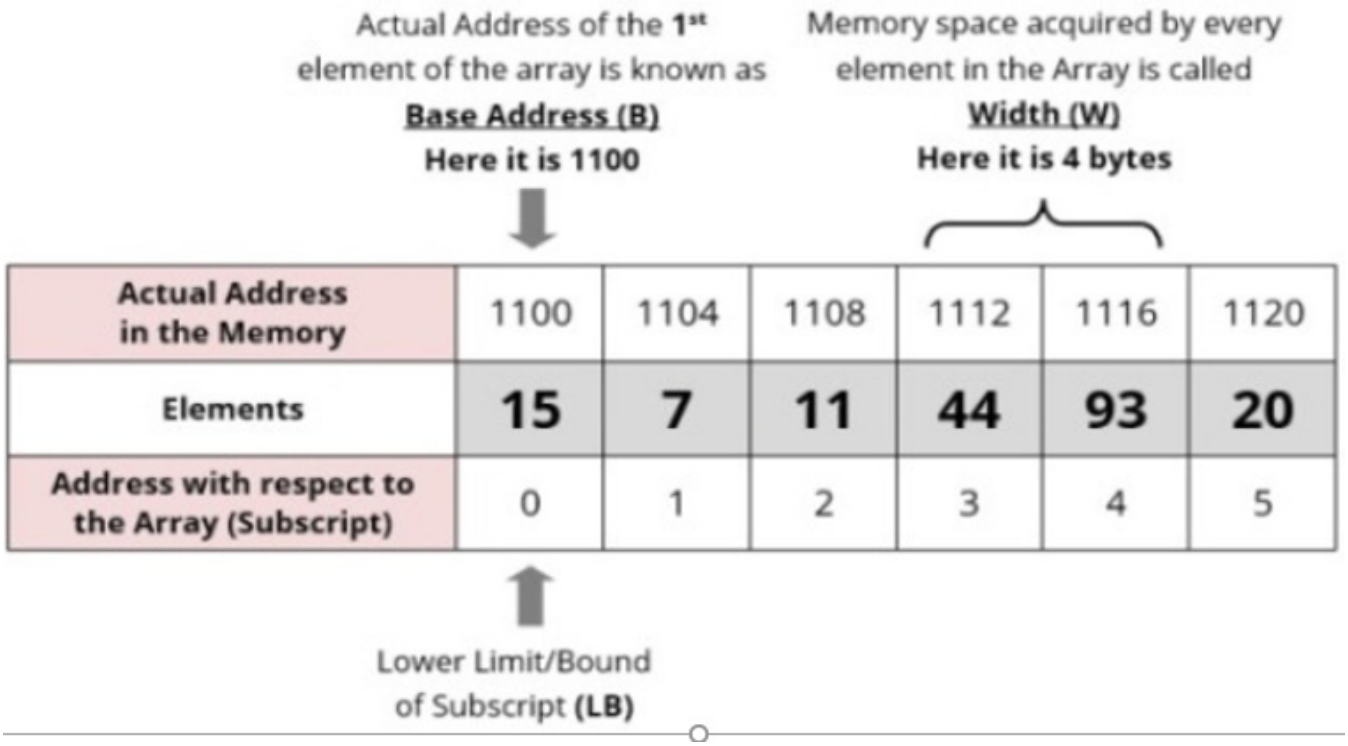
- बहुआयामी ऐरे का सरलतम रूप दो आयामी ऐरे है उदाहरणः

  int x (3) (4);

- उपरोक्त x एक दो आयामी (2डी) ऐरे है । ऐरे में 12 तत्व है। यहाँ x 3 पंक्ति के साथ तालिका के रूप में ऐरे है । और प्रत्येक पंक्ति में 4 स्तंभ है ।

	Column 1	Column 2	Column 3	Column 4
Row 1	x [0] [0]	x [0] [1]	x [0] [2]	x [0] [3]
Row 2	x [1] [0]	x [1] [1]	x [1] [2]	x [1] [3]
Row 3	x [2] [0]	x [2] [1]	x [2] [2]	x [2] [3]

एकल (एक) आयामी ऐरे में पता गणनाः



- एक ऐरे "A [I]" के एक तत्व की गणना निम्न सूत्र के उपयोग से करते हैं-

  Address of A [I] = B + W * (I - LB)

  Where,

  B = आधार पता

  W = ऐरे में उपस्थित एक तत्व की स्टोरेज साइज (बाइट में)

  I = जिस तत्व का पता ज्ञात करना है उसका सबस्क्रिप्ट

  LB = नीचली सीमा / उपलब्ध नहीं होने पर शून्य माने 0 (शून्य)

उदाहरण-

एक ऐरे [1300 - - - - 1900] का आधार पता 1020 और प्रत्येक तत्व का आकार मेमोरी में 2 बाइट्स के रूप में है । B[1700], का पता गणना कीजिए

हल-

दिए गए मान निम्न है B = 1020, LB = 1300, W = 2, I = 1700

A [I] का पता = B + W * (I - LB)

= 1020 + 2 * (1700-1300)

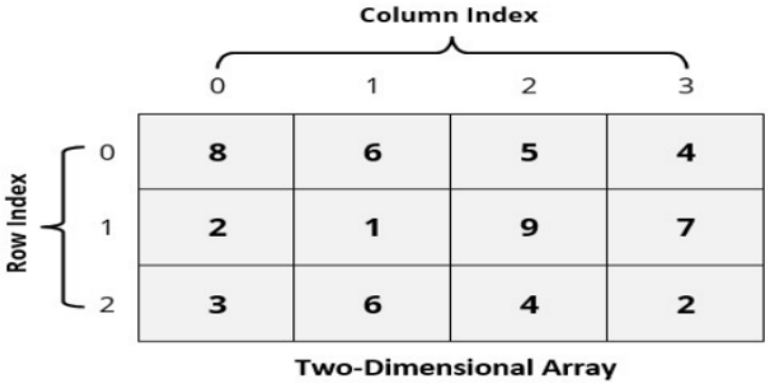
= 1020 + 2 * 400

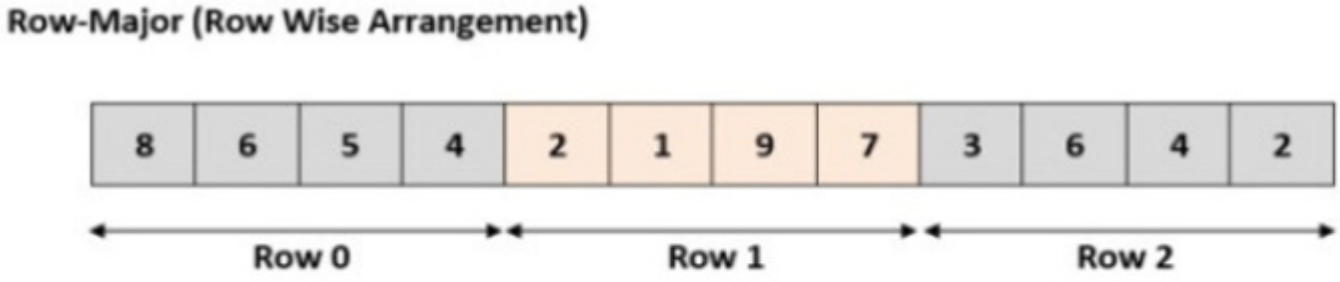
= 1020 + 800

= 1820 [Ans]

मल्टी (दो) आयामी ऐरे में पता गणना

- मेमोरी में एक 2-डी ऐरे के तत्वों को संग्रह करते समय इन्हें क्रमिक मेमोरी लोकेशन आवंटित किये जाते हैं। इसलिए उसके भंडारण को सक्षम करने के लिए2-डी ऐरे को लीनियराइज करते हैं। लीनियराइज करने के दो तरीके होते हैं। रो (पंक्ति) मेजर और कॉलम (स्तंभ) मेजर।





ऐरे के किसी तत्व "A [I] [J]" के पते की गणना नीचे दिए गए दो तरीके से की जा सकती है ।

(i) पंक्ति प्रमुख प्रणाली (Row Major System)

(ii) कॉलम प्रमुख प्रणाली (Column Major System)

पंक्ति प्रमुख प्रणाली

- पंक्ति प्रमुख प्रणाली में एक लोकेशन का पता निम्न सूत्र का उपयोग करके किया जाता है–

  A [I] [J] तत्व का पता = B + W * [N* (I - Lr) + (J - Lc)]

स्तंभ (कॉलम) प्रमुख प्रणाली

- कॉलम प्रमुख प्रणाली में एक लोकेशन का पता निम्न सूत्र का उपयोग करके किया जाता है:

  A [I] [J] तत्व का पता = B + W * [(I - Lr) + M* (J - Lc)]

  यहाँ पे,

  B = आधार पता

  I = जिस तत्व का पता ज्ञात करना है उसका पंक्ति सबस्क्रिप्ट

  J = जिस तत्व का पता ज्ञात करना है उसका स्तंभ सबस्क्रिप्ट

  W = ऐरे में उपस्थित एक तत्व की स्टोरेज साइज (बाइट मे)

  Lr = पंक्ति की नीचली सीमा / उपलब्ध नहीं होने पर शून्य माने 0 (शून्य)

  Lc = स्तंभ की नीचली सीमा / उपलब्य नहीं होने पर शून्य माने 0 (शून्य)

  M = मैट्रिक्स में पंक्तियों की संख्यां

  N = मैट्रिक्स में स्तंभों की संख्यां

ऐरे पर बुनियादी ऑपरेशन

निम्नलिखित ऑपरेशन ऐरे पर किये जा सकले है–

(i) ट्रेवर्सिंग (Traversing)– एक डाटा स्ट्रक्चर में मौजूद सभी डाटा तत्वों के प्रसंस्करण (प्रोसेसिंग) को ट्रेवर्सिंग कहते है ।

(ii) इनसर्शन (Insertion)– इनसर्शन का अर्थ एक डाटा स्ट्रक्चर में एक नये डाटा को जोडना ।

(iii) डिलिशन (deletion)– डिलिशन का अर्थ एक डाटा स्ट्रक्चर में एक डाटा तत्व को हटाना, यदि वह मौजूद है ।

(iv) अपडेट (Update)– दिए गए सूचकांक में एक तत्व अपठेट करता है ।