Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Recap of Arrays and Simultaneous Equations (S31 – S35)

Video Lecture Recap Quiz

Arrays and Simultaneous Equations

Q1) From the following scenario, choose the one which is more appropriate on the use of Array

A] to deal with large number of values.
B] to get sum of 2 numbers.
C] to get average of 3 numbers.
D] None of these

Q2) Let a = {5,9,12,34,56,43}, where a_1=5

Sum of (a_2+a_4+a_6) = ?

IIT Bombay

A] 21

B] 86

C] 73

D] None of these

Q3) Which of the following properties are true for arrays?

A] To store a large number of similar values

B] It is a collection of adjacent memory locations

C] An index could be used to refer to an individual element.

D] All of these

Q4) Choose appropriate answer for arrays in C++?

1. We must declare its name and size
2. Size could be float.

A] only 1 is true

B] only 2 is true

C] Both are true

D] Both are false

Q5) In C++, a[5]={2,5,6,3,2};
What will be the value corresponding to a[2*10-3*6] ?

A] 2

B] 3

C] 6

D] None of these

Q6) Consider the following code segment:

```
/* Assume "marks" is an array of type 'int'
      defined earlier */

int sum = 0;
for (int i =0; i < N; i = i +1) {
    sum = sum + marks[i];
}
int a = sum/N;
```

What does "a" represent?

A] Standard deviation

B] Average

C] sum

D] none

Q7) Which of the following are aspects of practical programming?

A] Check if the value given by user is out of bound

B] Display error message if value is out of bound

C] write explanatory comments

D] None of these

Recap Session

Arrays and Simultaneous Equations

Array in C++

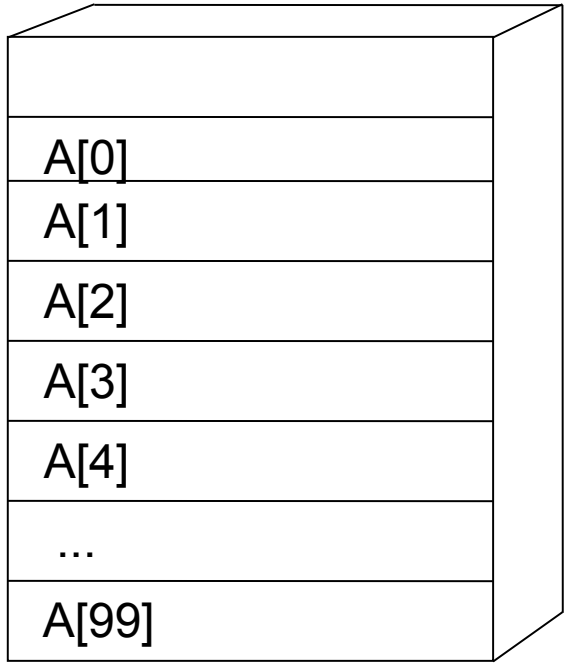
IIT Bombay

- Array is a collection of elements of the same type
  - It has a name, chosen by us, and a fixed size (number of elements)
- Declaring an array

  int marks[500], roll_numbers[500]; float distances[25];
- **Only one element** participates in an operation
  - Input or output
  - As an operand in an expression
  - As a location on LHS of an assignment statement

A pictorial view of C++ array

IIT Bombay



- Array A has 100 int elements

  int A[100];

- Suppose it stores 5 values

  53, 79, 41, 94, 38

A[0] = 53
A[1] = 79
A[2] = 41
A[3] = 94
A[4] = 38

Program to find sum of marks of N students

IIT Bombay

```
int main(){
   // program to find the sum of N marks
   int marks[600] , sum = 0, count, N;
   cin >> N;
   for (count =0; count < N; count = count +1){
         cin >> marks[count]; sum = sum + marks[count];
   }
   cout << sum;
return 0;
}
```

Multidimensional Array (Matrix)

- We can declare and use arrays with more than one dimension

  int A[50][40];

  - Declares a two dimensional array with 50 rows and 40 columns
- Each element is accessed by a reference requiring two index expressions, e.g.,

  A[i][j] = 3782;

  - Row index ‘i‘, can have a value from 0 to 49,
  - Column index ‘j’ can have a value from 0 to 39
- All rules for index expression, apply to index for each dimension

Simultaneous Equations

IIT Bombay

- Matrices are used to represent a system of simultaneous equations in multiple variables
- Consider the following equations in two variables

$$2x + 4y = 8 \qquad \text{eq.1}$$

$$4x + 3y = 1 \qquad \text{eq.2}$$

These equations can be represented as

$$\begin{pmatrix} 2 & 4 \\ 4 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 8 \\ 1 \end{pmatrix}$$

Simultaneous Equations …

IIT Bombay

- The Gaussian elimination technique essentially reduces the coefficient matrix to an upper triangular form:

$$\begin{pmatrix} 1 & a_{01} & a_{02} & \dots & a_{0n-1} \\ 0 & 1 & a_{12} & \dots & a_{1n-1} \\ 0 & 0 & 1 & & . \\ & & & & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ . \\ \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ . \\ \\ b_{n-1} \end{pmatrix}$$

System in upper triangular form

IIT Bombay

- When the coefficient matrix is reduced to the upper triangular form, we have the following system of equations

$$x[0] + a[0][1]\,x[1] + a[0][2]\,x[2] + \ldots + a[0][n-1]\,x[n-1] = b[0]$$

$$x[1] + a[1][2]\,x[2] + \ldots + a[1][n-1]\,x[n-1] = b[1]$$

$$\ldots$$

$$x[n-1] = b[n-1]$$

- Note that values of a[][] and b[] now, will be different from the original values
- Back substitution can be applied to calculate values of variables

Practice Problem1: Sieve Of Eratosthenes:

The Sieve of Eratosthenes is an ancient and famous algorithm to find the prime numbers in the range 2 to n (both 2 and n included).

As we know, prime numbers are numbers which are only divisible by one and itself.

We use this fact to solve the problem.

Algorithm:

1. We start with a list of consecutive integers from 2 to n: {2,3,4,...,n} (Assume all the elements are unmarked).
2. Mark the lowest unmarked element, p
3. Remove all multiples of the p in the list, except p itself
4. Repeat steps 2 & 3 until all elements are marked.

Algorithm (Contd.)

IIT Bombay

All elements left in list are prime numbers less than n.

Examples: Let us say n = 14

Input list: 2 3 4 5 6 7 8 9 10 11 12 13 14

Output list: 2 3 5 7 11 13

Algorithm (Contd.)

IIT Bombay

Step by step execution:

We mark 2 and remove 4,6,8,10,12,14

New list 2,3,5,7,9,11,13

Next we mark 3 and remove 9

New list 2,3,5,7,11,13

Now we mark 5,7,11,13 (nothing to remove as no multiples present) in consecutive steps

Algorithm (Contd.)

Finally our list contains:

2,3,5,7,9,11,13

We got the list of all primes from 2 to 14!

Code Outline:

IIT Bombay

```
#include<iostream>
using namespace std;
int main() {
int removed[100];
/*array of size 100 to keep track of all
removed      numbers
0 = not removed, 1 = removed */
```

Code Outline (Contd.)

IIT Bombay

```
//code to mark all elements with 0
code_snippet1

for( int i = 2; i < 100; i++) {
/* check if i is not removed then ‘remove’ all
multiples of i < 100 except i */

code_snippet2
}
```

Code Outline (Contd.)

IIT Bombay

```
//code to output all the primes < 100
code_snippet3
}
```

Fill in the three code snippets

Practice Problem2: Queue

IIT Bombay

Queue is an ordered list that supports only two operations, namely enqueue and dequeue.

Enqueue is an insertion operation that inserts the supplied element at the end of the queue.

Practice Problem2: Queue

IIT Bombay

Dequeue is a deletion operation that deletes the element at the start of the queue. The ends of the queue are called front and rear ends. Insertion happens at the rear end and deletion happens at the front end

Practice Problem2: Queue (contd…)

Initially queue is empty				
Enqueue(4)	4			
Enqueue(5)	4	5		
Enqueue(6)	4	5	6	
Dequeue()	5	6		
Enqueue(8)	5	6	8	
Enqueue(9)	5	6	8	9
Dequeue()	6	8	9	

Practice Problem2: Queue (contd…)

IIT Bombay

Question 1: Consider the queue 4, 6, 7, 8, 9 10. Perform the following operations on the queue and check your answers with your TA.

Enqueue(11)
Enqueue(13)
Dequeue()
Dequeue()
Dequeue()
Enqueue(12)

Practice Problem2: Queue (contd…)

IIT Bombay

Question 2
What is the minimum number of operations needed to remove 12 out of the queue?

Practice Problem2: Queue (contd…)

IIT Bombay

Question 3

Consider the following code which is an array implementation of queues. Consider the array to be of size 10. Consider the array to be global (which means can be accessed from anywhere in the program). Complete the function definitions of the enqueue and dequeue.

Practice Problem2: Queue (Code Snippet …)

IIT Bombay

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

int A[10];
int rear = -1; //stores the rear index
int front = -1; //stores the front index
```

Practice Problem2: Queue (Code Snippet …)


```
void enqueue(int ele) {
/* check if the queue is empty
* check if queue if full using conditions on rear and
* front(ask your TA if you can't!)
* if space available at last insert the element
* else shift all elements to start of array & insert the
new element */
code_snippet1
}
```

Practice Problem2: Queue (Code Snippet …)


```
void dequeue() {
/*
* check if queue is empty
* remove the front most element
*/
code_snippet2
}
```

Practice Problem2: Queue (Code Snippet …)


```
int main() {
//code snippet to create a queue : 1, 4, 6, 7
code_snippet3
return 0;
}
```