



## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

# Module 03: Programming in C++

## Arrays and Strings

Partha Pratim Das

Department of Computer Science and Engineering  
Indian Institute of Technology, Kharagpur

*ppd@cse.iitkgp.ernet.in*

Tanwi Mallick  
Srijoni Majumdar  
Himadri B G S Bhuyan



# Module Objectives

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

- Understand array usage in C and C++
- Understand vector usage in C++
- Understand `string` functions in C and `string` type in C++



# Module Outline

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

- Arrays and Vectors
  - Fixed size arrays – in C and C++
  - Arbitrary size arrays – in C and C++
  - vectors in C++
- Strings in C and C++
  - string functions in C and C++
  - string type in C++
  - String manipulation in C++



# Program 03.01: Fixed Size Array

## Module 03

Partha Pratim Das

Objectives & Outline

Arrays & Vectors

Fixed Size Array  
Arbitrary Size Array  
Vectors

Strings

Summary

### C Program

```
// File Name:Array_Fixed_Size.c:
#include <stdio.h>

int main() {
    short age[4];

    age[0] = 23;
    age[1] = 34;
    age[2] = 65;
    age[3] = 74;

    printf("%d ", age[0]);
    printf("%d ", age[1]);
    printf("%d ", age[2]);
    printf("%d ", age[3]);

    return 0;
}
```

23 34 65 74

### C++ Program

```
//FileName:Array_Fixed_Size_c++.cpp:
#include <iostream>

int main() {
    short age[4];

    age[0] = 23;
    age[1] = 34;
    age[2] = 65;
    age[3] = 74;

    std::cout << age[0] << " ";
    std::cout << age[1] << " ";
    std::cout << age[2] << " ";
    std::cout << age[3] << " ";

    return 0;
}
```

23 34 65 74

- No difference between arrays in C and C++



# Arbitrary Size Array

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

This can be implemented in C (C++) in the following ways:

- **Case 1:** Declaring a large array with size greater than the size given by users in all (most) of the cases
  - Hard-code the maximum size in code
  - Declare a manifest constant for the maximum size
- **Case 2:** Using `malloc` (`new[]`) to dynamically allocate space at run-time for the array



# Program 03.02: Fixed size large array in C

## Module 03

Partha Pratim Das

Objectives & Outline

Arrays & Vectors

Fixed Size Array  
Arbitrary Size Array  
Vectors

Strings

Summary

### Hard-coded

```
// FileName:Array_Large_Size.c:
#include <stdio.h>
#include <stdlib.h>

int main() {
    int arr[100], sum = 0, i;

    printf("Enter no. of elements: ");
    int count;
    scanf("%d", &count);

    for(i = 0; i < count; i++) {
        arr[i] = i;
        sum += arr[i];
    }
    printf("Array Sum: %d", sum);

    return 0;
}
```

Enter no. of elements: 10  
Array Sum: 45

- Hard-coded size

### Using manifest constant

```
// FileName:Array_Macro.c:
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

int main() {
    int arr[MAX], sum = 0, i;

    printf("Enter no. of elements: ");
    int count;
    scanf("%d", &count);

    for(i = 0; i < count; i++) {
        arr[i] = i;
        sum += arr[i];
    }
    printf("Array Sum: %d", sum);

    return 0;
}
```

Enter no. of elements: 10  
Array Sum: 45

- Size by manifest constant



# Program 03.03: Fixed large array / vector

## Module 03

Partha Pratim Das

Objectives & Outline

Arrays & Vectors

Fixed Size Array  
Arbitrary Size Array  
Vectors

Strings

Summary

### C (array & constant)

```
// FileName:Array_Macro.c:
#include <stdio.h>
#include <stdlib.h>

#define MAX 100

int main() {
    int arr[MAX], sum = 0, i;

    printf("Enter no. of elements: ");
    int count;
    scanf("%d", &count);
    for(i = 0; i < count; i++) {
        arr[i] = i;
        sum += arr[i];
    }
    printf("Array Sum: %d", sum);
    return 0;
}
```

Enter no. of elements: 10  
Array Sum: 45

- MAX is the declared size of array
- No header needed
- arr declared as int []

### C++ (vector & constant)

```
// FileName:Array_Macro_c++.cpp:
#include <iostream>
#include <vector>
using namespace std;
#define MAX 100

int main() {
    vector<int> arr(MAX); // Define-time size

    cout << "Enter the no. of elements: ";
    int count, j, sum = 0;
    cin >> count;
    for(int i = 0; i < count; i++) {
        arr[i] = i;
        sum += arr[i];
    }
    cout << "Array Sum: " << sum << endl;
    return 0;
}
```

Enter no. of elements: 10  
Array Sum: 45

- MAX is the declared size of vector
- Header vector included
- arr declared as vector<int>



# Program 03.04: Dynamically managed array size

## Module 03

Partha Pratim Das

Objectives & Outline

Arrays & Vectors

Fixed Size Array  
Arbitrary Size Array  
Vectors

Strings

Summary

### C Program

```
// FileName:Array_Malloc.c:
#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Enter no. of elements ");
    int count, sum = 0, i;
    scanf("%d", &count);

    int *arr = (int*) malloc
        (sizeof(int)*count);

    for(i = 0; i < count; i++) {
        arr[i] = i;
        sum += arr[i];
    }
    printf("Array Sum:%d ", sum);
    return 0;
}
```

Enter no. of elements: 10  
Array Sum: 45

- malloc allocates space using sizeof

### C++ Program

```
// FileName:Array_Resize_c++.cpp:
#include <iostream>
#include <vector>
using namespace std;

int main() {
    cout << "Enter the no. of elements: ";
    int count, j, sum=0;
    cin >> count;

    vector<int> arr;    // Default size
    arr.resize(count); // Set resize

    for(int i = 0; i < arr.size(); i++) {
        arr[i] = i;
        sum += arr[i];
    }
    cout << "Array Sum: " << sum << endl;
    return 0;
}
```

Enter no. of elements: 10  
Array Sum: 45

- resize fixes vector size at run-time





# Strings in C and C++

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

## String manipulations in C and C++:

- C-String and `string.h` library
  - C-String is an array of `char` terminated by `NULL`
  - C-String is supported by functions in `string.h` in C standard library
- `string` type in C++ standard library
  - `string` is a type
  - With operators (like `+` for concatenation) behaves like a built-in type



# Program 03.05: Concatenation of Strings

## Module 03

Partha Pratim Das

Objectives & Outline

Arrays & Vectors

Fixed Size Array  
Arbitrary Size Array  
Vectors

Strings

Summary

### C Program

```
// FileName:Add_strings.c:
#include <stdio.h>
#include <string.h>

int main() {
    char str1[] =
        {'H','E','L','L','O',' ',' ','\0'};
    char str2[] = "WORLD";

    char str[20];
    strcpy(str, str1);
    strcat(str, str2);

    printf("%s\n", str);

    return 0;
}
```

HELLO WORLD

- Need header string.h
- C-String is an array of characters
- String concatenation done with strcat function
- Need a copy into str
- str must be large to fit the result

### C++ Program

```
// FileName:Add_strings_c++.cpp:
#include <iostream>
#include <string>
using namespace std;

int main(void) {
    string str1 = "HELLO ";
    string str2 = "WORLD";

    string str = str1 + str2;

    cout << str;

    return 0;
}
```

HELLO WORLD

- Need header string
- string is a data-type in C++ standard library
- Strings are concatenated like addition of int



# More on Strings

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

Further,

- `operator=` can be used on strings in place of `strcpy` function in C.
- `operator<=`, `operator<`, `operator>=`, `operator>` operators can be used on strings in place of `strcmp` function in C



# Module Summary

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

- Working with variable sized arrays is more flexible with vectors in C++
- String operations are easier with C++ standard library



# Instructor and TAs

## Module 03

Partha Pratim  
Das

Objectives &  
Outline

Arrays &  
Vectors

Fixed Size Array  
Arbitrary Size  
Array  
Vectors

Strings

Summary

Name	Mail	Mobile
Partha Pratim Das, <i>Instructor</i>	ppd@cse.iitkgp.ernet.in	9830030880
Tanwi Mallick, <i>TA</i>	tanwimallick@gmail.com	9674277774
Srijoni Majumdar, <i>TA</i>	majumdarsrijoni@gmail.com	9674474267
Himadri B G S Bhuyan, <i>TA</i>	himadribhuyan@gmail.com	9438911655