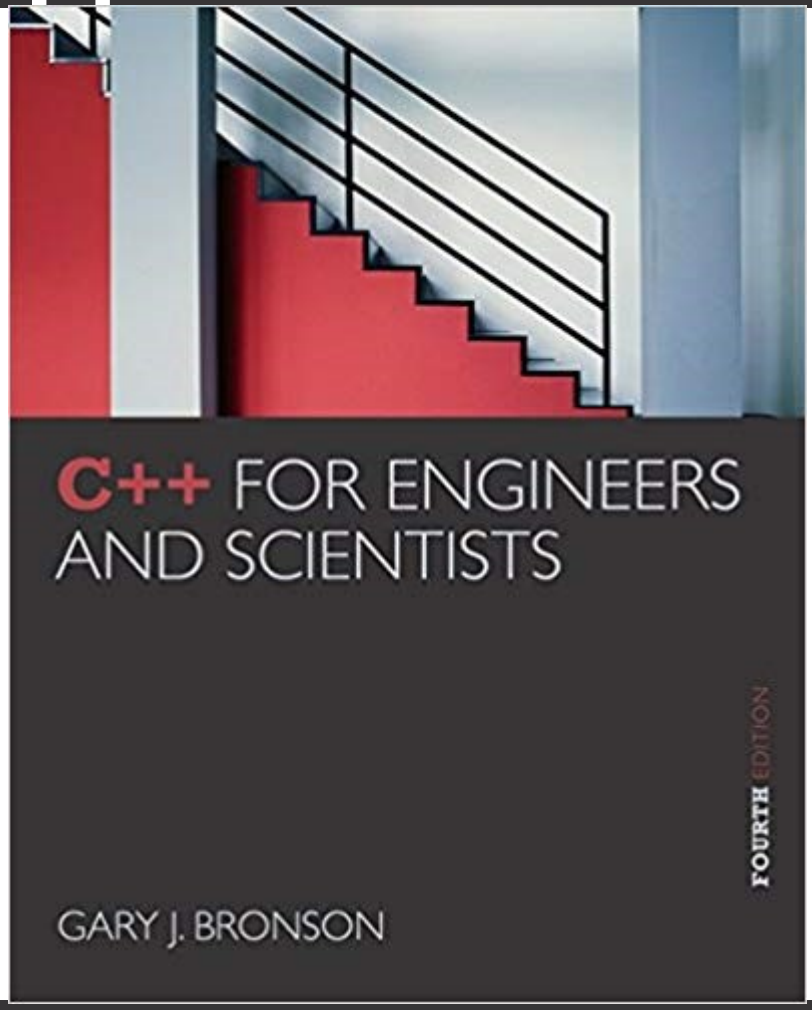
ELEG 1043
Computer Applications in Engineering

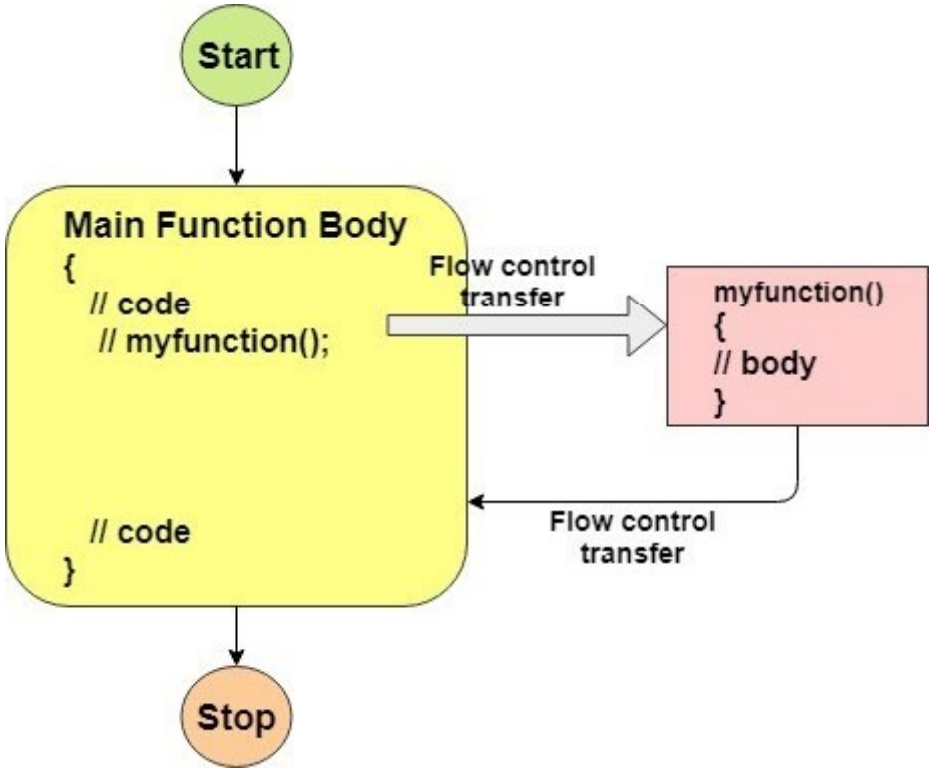


Lab Course 4

Acknowledgement

- Some of the slides or images are from various sources. The copyright of those materials belongs to their original owners.

Function


Start
Main Function Body
{
// code
// myfunction();
// code
}
Flow control transfer
myfunction()
{
// body
}
Flow control transfer
Stop

Exercise 1

- **Write two functions with Function Overloading Technique. One function is to add two integer numbers, and the other is to add two double numbers.**

Answer

```
#include <iostream>
using namespace std;
int add(int num1, int num2){
        int value = num1 + num2;
        return value;
}
double add(double num1, double num2){
        double value = num1 + num2;
        return value;
}
int main(){
int num1 = 1, num2 = 2;
cout<<add(num1, num2);
double dnum1 = 0.1, dnum2 = 0.3;
cout<<add(dnum1, dnum2);
}
```

Exercise 2

- **Write a function that accepts two numbers num1 and num2, calculates the difference between these two numbers. The difference, d, between two numbers is given by this formula (10 points):**

$$d = num1 - num2$$

Answer

```
#include <iostream>
using namespace std;
int diff(int num1, int num2){
        int value = num1 - num2;
        return value;
}
int main(){
int num1 = 1, num2 = 2;
cout<<diff(num1, num2);
}
```

Exercise 3

- **Write a program that is to build a function to judge if the number is the even number or odd number and if the number is odd and positive, display its value and the information " is a positive odd number.", where the number is received from keyboard and the main function calls this function (20 points).**

Answer

```
#include <iostream>
using namespace std;
void positiveOdd(int num){
        if(num%2 != 0 && num > 0)
        {
                cout<<num<<" is a positive odd"<<endl;
        }
}
int main(){
int num = 0;
cin>>num;
positiveOdd(num);
return 0;
}
```

Exercise 4

- **Bonus. Write a program that is to build a function to receive 20 numbers from the keyboard and display “You lose!” if the number is less than 10, and call this function in main function (10 points).**

Answer

```
#include <iostream>
using namespace std;
void game(){
        for( int i = 0; i < 20; i++)
        {
                int num = 0;
                cin>>num;
                if(num < 10)
                { cout<<“You lose!”<<endl;}
        }
}
int main(){
game()
return 0;
}
```

Chapter 7: Arrays

C++ FOR ENGINEERS AND SCIENTISTS

Case Study

- Arrays are useful in applications that require multiple passes through the same set of data elements
  - Statistical Analysis
  - Array: *X* = [98, 82, 67, 54, 78, 83, 95, 76, 68, 63]
  - Calculating
    - Mean value
    - Standard Deviation

Case Study

- Mean value

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N}$$

- Standard Deviation

$$\delta = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N-1}}$$

Mean value

```
double findAvg(int nums[], int numel)
{
    int i;
    double sumnums = 0.0;
    for (i = 0; i < numel; i++)
        sumnums = sumnums + nums[i];
    return (sumnums / numel);
}
```

Standard Deviation

```
double stdDev(int nums[], int numel, double avr)
{
    int i;
    double sumdevs = 0.0;
    for (i = 0; i < numel; i++)
        sumdevs = sumdevs + pow((nums[i] – avr),2);
    return (sqrt(sumdevs/(numel – 1.0)));
}
```

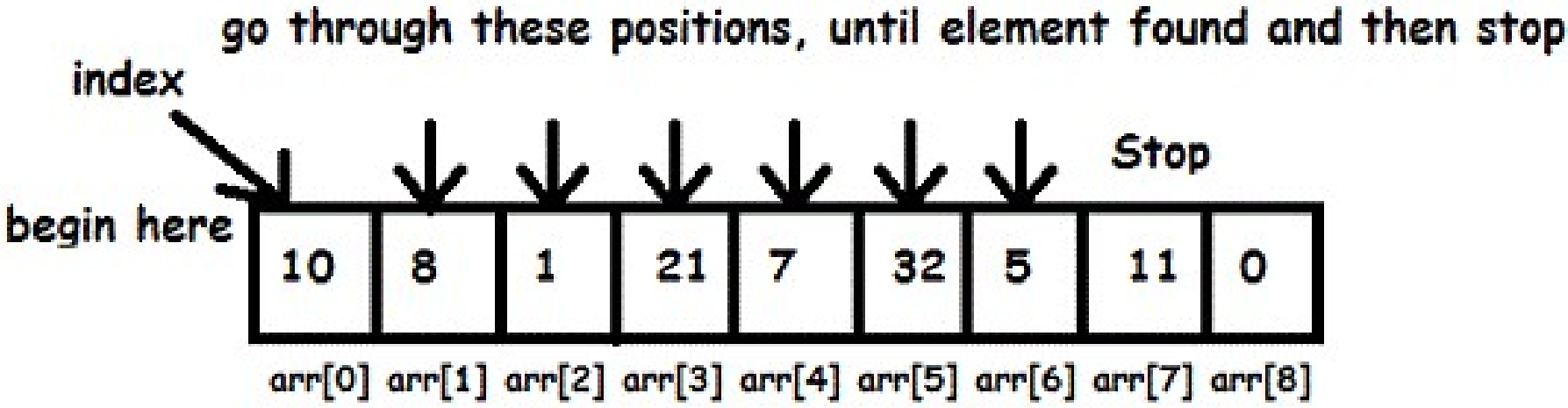
Main Function

```
#include <iosteam>
using namespace std;
int main(){
    const int NUMELS = 10;
    int values[NUMELS] = {98, 82, 67, 54, 78, 83, 95, 76, 68, 63};
    double average, sDev;
    average = findAvg(values, NUMELS); // call the function
    sDev = stdDev(values, NUMELS, average); // call the function
    cout << "The average of the numbers is “<<average << endl;
    cout << "The standard deviation of the numbers is “<<sDev << endl;
    return 0;
}
```

Linear Search

- Each item in the list is examined in the order in which it occurs
- Not a very efficient method for searching
- Advantage is that the list does not have to be in sorted order

Linear Search (continued)



Element to search : 5

Linear Search (continued)

```
#include <iostream>
using namespace std;
/* Linear Search Function */
int linear_search(int arr[], int length, int val);

int main(){
int arr[5] = {3,7,10,6,9};
int val = 6;
cout<<“The index of ”<<val <<“ int the array is ”<<
linear_search(arr, 5, val)<<endl;
return 0;}
```

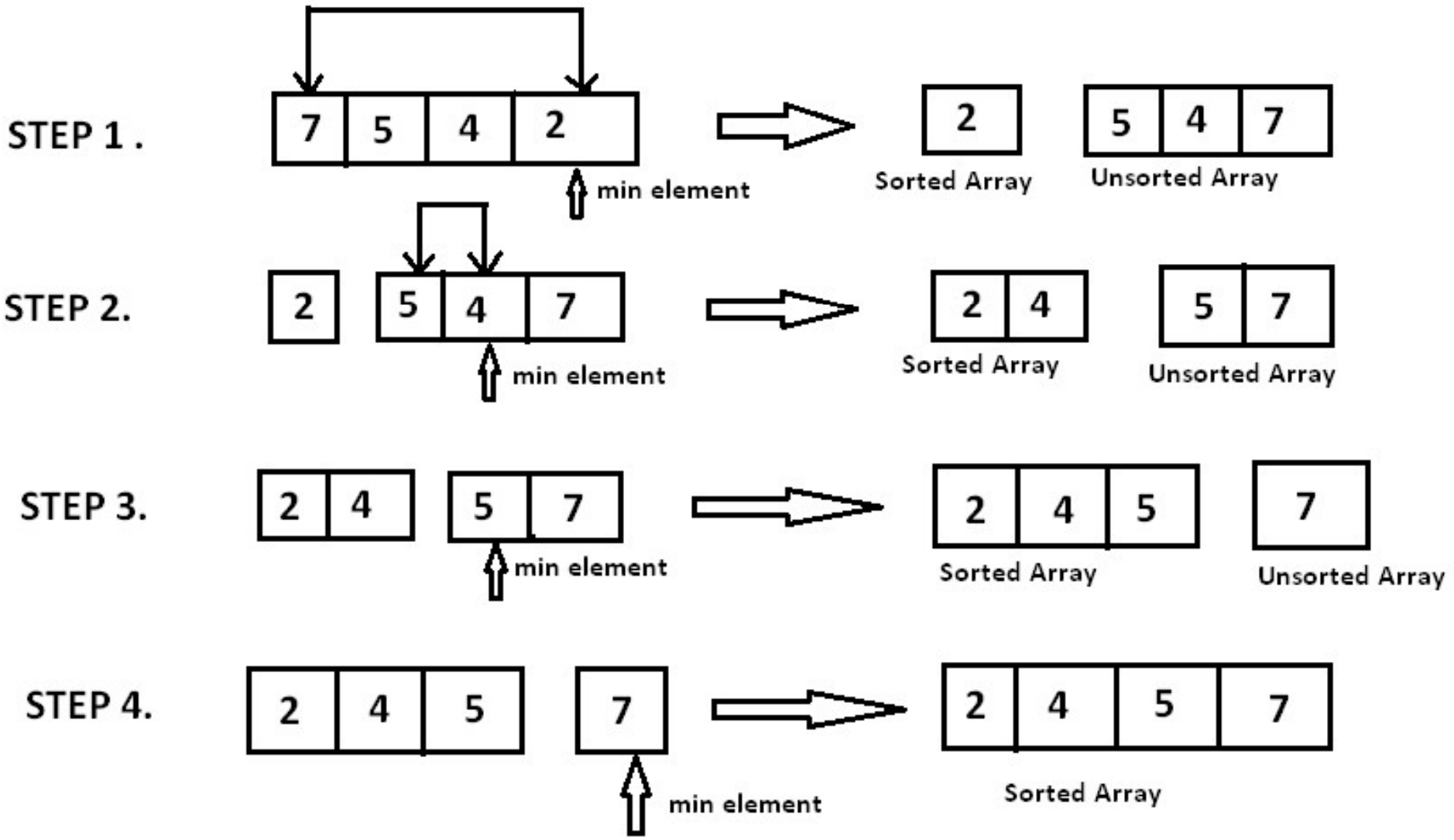
Linear Search (continued)

```
/* Linear Search Function */
int linear_search(int arr[], int length, int val)
{
    int key = -1;
    for (int i = 0; i < length; i++)
    {
     if (arr[i] == val)
     { key = i; break;}
    }
    return key;
}
```

Selection Sort

- Smallest element is found and exchanged with the first element
- Next smallest element is found and exchanged with the second element
- Process continues *n*-1 times, with each pass requiring one less comparison

Selection Sort (continued)



Selection Sort (continued)

```
#include <iostream>
using namespace std;
void selectionSort(int arr[], int length)
int main(){
    int arr[5] = {5,4,3,9,6};
    for(int i = 0; i < 5; i++)
    {cout<<arr[i] <<" ";}
    cout<<endl;
    selectionSort(arr, 5);
    for(int i = 0; i < 5; i++)
    {cout<<arr[i] <<" ";}
    cout<<endl;
    cout<<diff(3,2)<<endl;
    return 0;}
```

Selection Sort (continued)

```
void selectionSort(int arr[], int length)
{
	for(int i = 0; i < length; i++)
	{
		int min = arr[i];
		int minIndex = i;
		for(int j = i; j < length; j++)
		{
		  if(min > arr[j])
		  {
			min = arr[j];
			minIndex = j;
		  }
		}
		cout<<minIndex<<endl;
		int tmp = arr[minIndex];
		arr[minIndex] = arr[i];
		arr[i] = tmp;
	}
}
```