**Arrays**

1. Write a program which reads in 10 integers from the user and stores them in an array. Find the largest value in the array and print it.
2. Modify the last program to use a preprocessor constant for the size of the array and in the test condition of the loop which processes the array.
3. Modify the last program to find mean of *n* numbers using arrays.
4. Write a program to interchange the largest and the smallest number in the array.
5. Write a program to find the second biggest number using an array of *n* numbers.
6. Write a program to find whether the array of integers contain a duplicate number. If it’s there print the position of duplicate numbers.
7. Write a program which can store 10 integers in an array. Fill the array with “random” numbers using the library functions **rand()** and **srand()** instead of reading them from the user. Find the largest element in the array and print it out.

To use the random number generator, first call **srand()** like this:

**srand(17)**;

to “seed” the random number generator. Passing different values to **srand()** will make **rand()** return a different sequence of values.

Each time **rand()** is called it returns a “random” integer. Use the mod operator ( % ) to get a value in the desired range. For example:

int **result;**

result = **rand() % 1000;**

will assign a random value in the range **0 – 999** to the variable **result.**

Make sure your program contains the line:

**#include <stdlib.h>**

to include information about the **rand()** and **srand()** functions.

1. Modify the last program so that instead of finding the largest element in the array, the program sorts the elements of the array into ascending order.

**Strings**

**Reading Strings**

If we declare a string by writing char str[50];

Then str can be read from the user by using three ways:

1. Using scanf() function

2. Using gets() function

3. Using getchar() function repeatedly

1. Write program to read a string in the above three ways.

**Writing strings**

1. The string can be displayed on screen using three ways:

1. Using printf() function

2. Using puts() function

3. Using putchar()function repeatedly.

1. Modify the above program to display the read string.
2. 3. Run the following program and analyze the result.

#include<stdio.h>

main()

{

char str = “Hello”;

printf(“\n %s”,str);

printf(“\n %s”,&str);

printf(“\n%s”,&str[2]);

}

1. 4. Run the following program and analyze the result. It’s about the use of width

and precision specifications along with %s .

#include<stdio.h>

main()

{

char str[] = “Introduction to C”;

printf(“\n |%s|”,str);

printf(“\n |%20s|”,str);

printf(“\n |%-20s|”,str);

printf(“\n |%.4s|”,str);

printf(“\n |%20.4s|”,str);

printf(“\n |%-20.4s|”,str);

}

1. Write a program to find the length of a string.
2. Write a program to convert characters of a string to upper case.

*Note: Recall that ASCII code for A-Z varies from 65 to 91 and the ASCII code for a-z*

*ranges from 97 to 123*

1. Write a program to concatenate two strings. (Do the same operation using the string
2. library function *concat()* and analyze the behavior; you should include *string .h*)

. Write a program to compare two strings. (Do the same operation using the string

library function *strcmp()* and analyze the behavior; you should include *string .h*)

1. Write a program to check whether the entered string is a palindrome.