PROGRAMMING FOR PROBLEM SOLVING LAB (ES 351 CS)

LAB Manual AY 2020-21

1. (a) Finding maximum and minimum of given set of numbers.

Aim: To write a C-Program for finding the maximum and minimum of given set of numbers.

Program:

```
#include<stdio.h>
int main(void)
{
    int arr[20];
    int i, max, min, size;
    printf("Enter size of the array:");
    scanf("%d", &size);
    printf("Enter %d elements into the array:", size);
    for(i=0; i<size; i++)
        scanf("%d", &arr[i]);

    /* assume the first element as maximum and minimum */
    max = arr[0];
    min = arr[0];
    for(i=1; i<size; i++)
    {
        /* If current element of array is greater than max */
        if(arr[i]>max)
            max = arr[i];
        /* If current element of array is smaller than min */
        if(arr[i]<min)
            min = arr[i];
    }
    printf("Maximum element = %d\n", max);
    printf("Minimum element = %d\n", min);
    getch();
    return 0;
}
```

Output:

```
Enter size of the array:5
Enter 5 elements into the array:3 5 7 1 9
Maximum element = 9
Minimum element = 1
```

1. (b) Finding roots of quadratic equation.

Aim: To write a C-Program for finding roots of quadratic equation.

Program:

```
#include<stdio.h>
#include<math.h>
int main(void)
{
        float a,b,c,disc,root1,root2;
        printf("Enter a, b, c values:");
        scanf("%f %f %f",&a,&b,&c);
        disc=b*b-4*a*c;
        if(disc>0)
        {
                printf("\n****ROOTS REAL & UN EQUAL****\n");
                root1=(-b+sqrt(disc))/(2*a);
                root2=(-b-sqrt(disc))/(2*a);
                printf("Root1=%f & Root2=%f",root1,root2);
        }
        else if(disc==0)
        {
                printf("\n****ROOTS REAL & EQUAL****\n");
                root1=-b/(2*a);
                root2=-b/(2*a);
                printf("Root1=%f & Root2=%f",root1,root2);
        }
        else
                printf("\n****IMAGINARY ROOTS****\n");
        return 0;
}
```

Output:

```
        Enter a,b,c values:2 9 -5
        ****ROOTS REAL & UN EQUAL****
        Root1=0.500000 & Root2=-5.000000
        ------------------------------------------------------
        Enter a,b,c values:1 -12 36
        ****ROOTS REAL & EQUAL****
        Root1=6.000000 & Root2=6.000000
        ------------------------------------------------------
        Enter a,b,c values:1 -10 34
        ****IMAGINARY ROOTS****
```

Example

Find the roots of quadratic equation 2x2+9x-5=0

a=2, b=9 and c=-5

disc = 9*9-4*2*-5 = 121

∵ disc > 0 roots are real and unequal

root1= (-b+√disc)/2*a = (-9+√121)/(2*2) = (-9+11)/4 = 2/4 = 0.5
root2= (-b-√disc)/2*a = (-9-√121)/(2*2) = (-9-11)/4 = -20/4 = -5
2. Find the roots of quadratic equation x2

-12x+36=0

a=1, b=-12 and c=36
disc = (-12*-12)-4*1*36 = 0
∵ disc = 0 roots are real and equal
root1= -b/2*a = -(-12)/(2*1) = 12/2 = 6
root2= -b/2*a = -(-12)/(2*1) = 12/2 = 6
3. Find the roots of quadratic equation x2

-10x+34=0

a=1, b=-10 and c=34
disc = 10*10-4*1*34 = -36
∵ disc < 0 Roots are imaginary

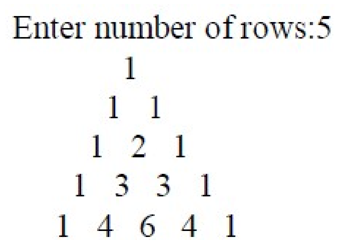
2. (a) Generating Pascal triangle.

Aim: To write a C-Program for generating Pascal Triangle.

Program:

```
#include<stdio.h>
int main(void)
{
	int rows,k=1,space,i,j;
	system("cls"); //clrscr();
	printf("Enter number of rows: ");
	scanf("%d",&rows);

	for(i=0;i<rows;i++)
	{
		for(space=1; space <= rows-i; space++)
			printf(" ");
		for(j=0;j<=i;j++)
		{
			if(j == 0||i == 0)
				k=1;
			else
				k=k*(i-j+1)/j;
			printf("%4d",k);
		}
		printf("\n");
	}
	getch();
	return 0;
}
```

Output:



```
Enter number of rows:5
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
```

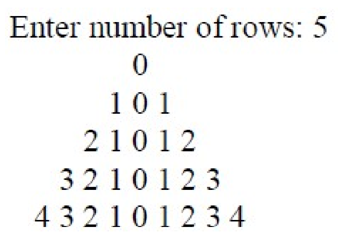
2. (b) Pyramid of numbers

Aim: To write a C-Program for generating Pyramid of numbers.

Program:

```
#include <stdio.h>
#include <math.h>
int main()
{
        int rows,space,i,j;
        system("cls"); //clrscr();
        printf("Enter number of rows: ");
        scanf("%d",&rows);

        for(i=0; i<rows; i++)
        {
                for(space=1; space <= rows-i; space++)
                printf(" ");
                for(j=0-i; j <= i; j++)
                {
                        printf("%2d",abs(j));
                }
                printf("\n");
        }
        getch();
        return 0;
}
```

Output:



```
Enter number of rows: 5
         0
       1 0 1
     2 1 0 1 2
   3 2 1 0 1 2 3
 4 3 2 1 0 1 2 3 4
```

3. (a) Recursion: Factorial.

Aim: To write a C-Program for finding factorial of a given number using recursion.

Program:

```
#include<stdio.h>
int factorial(int);
int main(void)
{
        int n,res;
        system("cls"); //clrscr();
        printf("Enter any number:");
        scanf("%d",&n);
        res=factorial(n);
        printf("Factorial of %d is: %d", n, res);
        getch();
        return 0;
}

int factorial(int num)
{
        if(num == 0)
                return 1;
        else
                return (num * factorial(num-1));
}
```

Output:

```
Enter any number:6
Factorial of 6 is: 720
```

3. (b) Recursion: Fibonacci.

Aim: To write a C-Program for printing Fibonacci sequence up to the given number of terms using recursion.

Program:

```
#include <stdio.h>
int fibonacci(int);
int main(void)
{
        int nterms, fib = 0, i;
        system("cls"); //clrscr();
        printf("Enter the number of terms:");
        scanf("%d", &nterms);

        printf("Fibonacci series terms are:\n");
        for (i = 0; i < nterms; i++)
        {
                printf("%d, ", fibonacci(fib));
                fib++;
        }
        getch();
        return 0;
}

int fibonacci(int n)
{
        if(n == 0 || n == 1)
                return n;
        return (fibonacci(n - 1) + fibonacci(n - 2));
}
```

Output:

```
Enter the number of terms:6
Fibonacci series terms are:
0, 1, 1, 2, 3, 5,
```

3. (c) Recursion: GCD.

Aim: To write a C-Program for finding GCD of a given numbers using recursion.

Program:

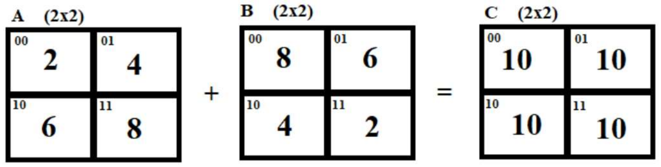
```
#include <stdio.h>
int gcd(int, int);
int main(void)
{
        int n1, n2;
        system("cls"); //clrscr();
        printf("Enter two integers: ");
        scanf("%d %d", &n1, &n2);
        printf("G.C.D of %d and %d is %d.", n1, n2, gcd(n1,n2));
        getch();
        return 0;
}

int gcd(int n1, int n2)
{
        if (n2 != 0)
                return gcd(n2, n1%n2);
        else
                return n1;
}
```

Output:

```
Enter two integers: 6 24
G.C.D of 6 and 24 is 6.
```

4. (a) Matrix addition using arrays.

Aim: To write a C-Program for matrix addition using arrays.



Program:

```
#include<stdio.h>
int main(void)
{
        int a[10][10],b[10][10],c[10][10],rows,cols,i,j;
        system("cls"); //clrscr();

        printf("Enter Number of Rows[<10]: ");
        scanf("%d",&rows);
        printf("Enter Number of Columns[<10]: ");
        scanf("%d",&cols);

        printf("Enter A(%dx%d) matrix: ", rows, cols);
        for(i=0;i<rows;i++)
                for(j=0;j<cols;j++)
                        scanf("%d",&a[i][j]);

        printf("Enter B(%dx%d) matrix: ", rows, cols);
        for(i=0;i<rows;i++)
                for(j=0;j<cols;j++)
                        scanf("%d",&b[i][j]);

        for(i=0;i<rows;i++)
                for(j=0;j<cols;j++)
                        c[i][j]=a[i][j]+b[i][j];

        printf("Resultant matrix C(%dx%d):\n", rows, cols);
        for(i=0;i<rows;i++)
        {
                for(j=0;j<cols;j++)
                        printf("%d ",c[i][j]);
                printf("\n");
        }
        getch();
        return 0;
}
```

Output:

```
Enter Number of Rows[<10]: 2
Enter Number of Columns[<10]: 2
Enter A(2x2) matrix: 2 4 6 8
Enter B(2x2) matrix: 8 6 4 2
The resultant matrix C(2x2):
10 10
10 10
```

4. (b) Matrix multiplication using arrays.

Aim: To write a C-Program for matrix multiplication using arrays.

Program:

```
#include<stdio.h>
int main(void)
{
        int a[10][10], b[10][10], c[10][10], rows, cols, i, j, k;
        system("cls"); //clrscr();

        printf("\nEnter Number of rows:");
        scanf("%d",&rows);
        printf("Enter Number of columns:");
        scanf("%d",&cols);

        printf("Enter A(%dx%d) matrix: ", rows, cols);
        for(i=0;i<rows;i++)
                for(j=0;j<cols;j++)
                        scanf("%d", &a[i][j]);

        printf("Enter B(%dx%d) matrix: ", rows, cols);
        for(i=0;i<rows;i++)
                for(j=0;j<cols;j++)
                        scanf("%d", &b[i][j]);

        for(i=0;i<rows;i++)
                for(j=0;j<cols;j++)
                {
                        c[i][j]=0;
                        for(k=0;k<cols;k++)
                                c[i][j]=c[i][j]+a[i][k]*b[k][j];
                }

        printf("\nResultant Matrix C(%dx%d):\n",rows,cols);
        for(i=0;i<rows;i++)
        {
                for(j=0;j<cols;j++)
                        printf("%d ", c[i][j]);
                printf("\n");
        }
        getch();
        return 0;
}
```

Output:

```
Enter Number of rows:2
Enter Number of columns:2
Enter A(2x2) matrix: 3 2 1 2
Enter B(2x2) matrix: 3 2 1 4

Resultant Matrix C(2x2):
11 14
5 10
```

5. (a) Linear Search.

Aim: To write a C-Program for Linear Search.

Program:

```
#include<stdio.h>
int seqSearch(int[],int,int);
int main(void)
{
        int list[20], target, index, i, size;
        system("cls"); //clrscr();
        printf("Enter the size of the list:");
        scanf("%d", &size);
        printf("Enter any %d elements:\n", size);

        for (i = 0; i < size; i++)
                scanf("%d", &list[i]);
        printf("Enter a target value to search:");
        scanf("%d", &target);
        index=seqSearch(list,target,size);

        if (index==-1)
                printf("%d isn't present in the list.\n", target);
        else
                printf("%d is present at %d in the list",target, index);
        getch();
        return 0;
}

int seqSearch(int list[],int target,int size)
{
        int index;
        for (index = 0; index < size; index++)
```

```
                if (list[index] == target)
                        return index;
        return -1;
}
```

Output:

```
  Enter the size of the list:6
  Enter any 6 elements:
  2 4 1 3 5 9
  Enter a target value to search:5
  5 is present at 4 in the list
```

5. (b) Binary search.

Aim: To write a C-Program for Binary search.

Program:

```
#include <stdio.h>
int binSearch(int[],int,int);
int main(void)
{
        int i, index, size, target, list[20];
        system("cls"); //clrscr();
        printf("Enter size of the list:");
        scanf("%d",&size);
        printf("Enter any %d elements:\n", size);
        for(i = 0; i < size; i++)
                scanf("%d", &list[i]);

        printf("Enter target value:\n");
        scanf("%d", &target);
        index=binSearch(list,target,size);

        if(index==-1)
                printf("%d isn't found in the list",target);
        else
                printf("%d is fount at %d in the list",target,index);
        getch();
        return 0;
}

int binSearch(int list[], int target, int size)
{
        int first, mid, last;
        first = 0;
        last = size - 1;
        while (first <= last)
        {
```

```
		mid = (first+last)/2;
		if(target == list[mid])
			return mid;
		else if (target > list[mid])
			first = mid+1;
		else
			last = mid-1;
	}
	return -1;
}
```

Output:

```
Enter size of the list:6
Enter any 6 elements:
3 7 2 9 12 34
Enter target value:
9
9 is fount at 3 in the list
```

6. (a) Bubble Sort.

Aim: To write a C-Program for Bubble Sort.

Program:

```
#include <stdio.h>
void swap(int*, int*);
void bubbleSort(int[],int);
void printArray(int[],int);
int main(void)
{
	int list[] = {64, 34, 25, 12, 22, 11, 90};
	int size = sizeof(list)/sizeof(list[0]);
	bubbleSort(list, size);
	printf("Sorted array(Bubble Sort): \n");
	printArray(list, size);
	getch();
	return 0;
}

void swap(int *xp, int *yp)
{
	int temp = *xp;
	*xp = *yp;
	*yp = temp;
}
```

```
// A function to implement bubble sort
void bubbleSort(int list[], int size)
{
        int cur, walk;
        for (cur = 0; cur < size; cur++)
                for (walk = size-1; walk > cur; walk--)
                        if (list[walk] < list[walk-1])
                                swap(&list[walk-1], &list[walk]);
}

/* Function to print an array */
void printArray(int list[], int size)
{
        int i;
        for (i=0; i < size; i++)
                printf("%d ", list[i]);
}
```

Output:

```
Sorted array(Bubble Sort):
11 12 22 25 34 64 90
```

6. (b) Selection Sort.

Aim: To write a C-Program for Selection Sort.

Program:

```
#include <stdio.h>
void swap(int*,int*);
void selectionSort(int[],int);
void printArray(int[],int);
// Driver program to test above functions
int main(void)
{
        int list[] = {64, 25, 12, 22, 11};
        int size = sizeof(list)/sizeof(list[0]);
        selectionSort(list, size-1);
        printf("Sorted array(Selection Sort): \n");
        printArray(list, size);
        getch();
        return 0;
}

void swap(int* xp, int* yp)
{
        int temp = *xp;
        *xp = *yp;
        *yp = temp;
}
```

```
void selectionSort(int list[], int size)
{
	int cur, walk, smlst;
	// One by one move boundary of unsorted subarray
	for (cur = 0; cur < size; cur++)
	{
		// Find the minimum element in unsorted array
		smlst = cur;
		for (walk = cur+1; walk <= size; walk++)
			if (list[walk] < list[smlst])
				smlst = walk;
		// Swap the found minimum element with the first element
		swap(&list[smlst], &list[cur]);
	}
}

/* Function to print the list */
void printArray(int list[], int size)
{
	int i;
	for (i=0; i < size; i++)
		printf("%d ", list[i]);
	printf("\n");
}
```

Output:

```
Sorted array(Selection Sort):
11 12 22 25 64
```

7. Functions for string manipulations.

Aim: To write a C-Program for string manipulations functions.

Program:

```
//Program for String Manipulation Functions
#include<stdio.h>
#include<String.h>
int main(void)
{
	char str1[20], str2[20], str3[20], str4[20];
	int len,i,j,k;
	system("cls"); //clrscr();

	puts("Enter string1[<20]:");
	gets(str1);
```

```
        puts("Enter string2[<20]:");
        gets(str2);

        //Finding the length using strlen() function
        printf("***String Lenght Functions***\n");
        len=strlen(str1);
        printf("String1 length=%d\n",len);
        len=strlen(str2);
        printf("String2 length=%d\n",len);

        //Comparing the strings using strcmp() and strncmp() functions
        printf("***String comparison***\n");
        i=strcmp(str2,str1);
        j=strncmp(str2,str1,3);
        printf("i=%d j=%d\n",i,j);

        //Copying the strings using strcpy() and strncpy() functions
        printf("***String copying***\n");
        strcpy(str3,str1);
        puts(str3);
        strncpy(str4,str1,3);
        puts(str4);

        //Concatenating the strings using strcat() and strncat() functions
        printf("***String Concatenation***\n");
        strcat(str3,str1);
        puts(str3);
        strncat(str4,str1,3);
        puts(str4);

        //Reversing the strings using strrev() function
        printf("***String Reverse***\n");
        strrev(str1);
        puts(str1);

        getch();
        return 0;
}
```

Output:

```
    Enter string1[<20]:
    hyderabad
    Enter string2[<20]:
    hydarabad
    ***String Lenght Functions***
    String1 length=9
    String2 length=9
```

```
***String comparison***
i=-1 j=0
***String copying***
hyderabad
hyd
***String Concatenation***
hyderabadhyderabad
hydhyd
***String Reverse***
dabaredyh
```

8. (a) Programs on structures.

Aim: To write a C-Program for implementing structures concept.

Program:

```
#include<stdio.h>

typedef struct
{
        int ht_no,m1,m2,m3;
        char sname[20];
}STUDENT;

int main(void)
{
        STUDENT std;
        system("cls"); //clrscr();
        printf("Enter the Hall-Ticket No., Name of the student and 3-Subject Marks:\n");
        scanf("%d %s %d %d %d",&std.ht_no,std.sname,&std.m1,&std.m2,&std.m3);
        printf("Hall-Ticket no of the student is: %d\n",std.ht_no);
        printf("Name of the student is: %s\n",std.sname);
        printf("Marks in Maths is: %d\n",std.m1);
        printf("Marks in Physics is: %d\n",std.m2);
        printf("Marks in Computer Science is: %d\n",std.m3);
        getch();
        return 0;
}
```

Output:

```
Enter the Hall-Ticket No., Name of the student and 3-Subject Marks:
101
Azher
20
18
18
Hall-Ticket no of the student is: 101
```

```
Name of the student is: Azher
Marks in Maths is: 20
Marks in Physics is: 18
Marks in Computer Science is: 18
```

8. (b) Programs on unions.

Aim: To write a C-Program for implementing unions concept.

Program:

```
#include <stdio.h>
#include <string.h>

union STUDENT
{
        char name[20];
        char semister[20];
        float average;
};

int main(void)
{
        union STUDENT std1;
        union STUDENT std2;
        // Assigning values to std1 union variable
        strcpy(std1.name,"Asher");
        strcpy(std1.semister, "First");
        std1.average = 86.50;
        printf("***Student-1 Details***\n");
        printf("Name : %s \n", std1.name);
        printf("Semister : %s \n", std1.semister);
        printf("Average : %f \n\n", std1.average);
        // Assigning values to std2 union variable
        printf("***Student-2 Details***\n");
        strcpy(std2.name, "Azhar");
        printf("Name : %s \n", std2.name);
        strcpy(std2.semister, "First");
        printf("Semister : %s \n", std2.semister);
        std2.average = 99.50;
        printf("Average : %f \n", std2.average);
        getch();
        return 0;
}
```

Output:

```
***Student-1 Details***
Name :
Semister :
Average : 86.500000
```

```
***Student-2 Details***
Name : Azhar
Semister : First
Average : 99.500000
```

9. Finding the number of characters, words and lines of given text file.

Aim: To write a C-Program for Finding the number of characters, words and lines of given text file.

Program:

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
FILE* fp;
char fname[100];
char ch;
int characters, words, lines;
printf("Enter source File Name: ");
scanf("%s", fname);
fp = fopen(fname, "r");
if (fp == NULL)
{
printf("\nUnable to open file...\n");
printf("Please check the file exists and have read privilege...\n");
exit(1);
}
/*Logic to count characters, words and lines.*/
characters = words = lines = 0;
while ((ch = fgetc(fp)) != EOF)
{
//Counting Characters
characters++;
//Counting Lines
if (ch == '\n' || ch == '\0')
lines++;
//Counting Words
if (ch == ' ' || ch == '\t' || ch == '\n' || ch == '\0')
words++;
}
// Printing File Statistics
printf("\nTotal characters = %d", characters);
printf("\nTotal words = %d", words);
printf("\nTotal lines = %d", lines);
// Close files to release resources
fclose(fp);
getch();
return 0;
}
```

sample.txt

```
I love programming.
Working with files in C programming is fun.
I am learning C programming.
```

Output:

```
Enter source File Name: sample.txt
Total characters = 107
Total words = 18
Total lines = 3
```