Laboratory Manual

For

C-Programming -II

(CT 215)

B.Tech (IT)

SEM II

June 2015

Faculty of Technology
Dharmsinh Desai University
Nadiad.
www.ddu.ac.in

Table of Contents

EXPERIMENT-1
Implement the programs of Arrays.....07

EXPERIMENT-2
Implement the programs of Character Array.....08

EXPERIMENT-3
Implement the programs using Functions.....09

EXPERIMENT-4
Implement programs using Recursive function & Arrays as arguments to function 10

EXPERIMENT-5
Implement the programs using Structures.....11

EXPERIMENT-6
Implement the programs using Union.....12

EXPERIMENT-7
Implement the programs using Pointers.....13

EXPERIMENT-8
Implement programs for Files & Command line arguments.....14

EXPERIMENT-9
Implement programs for Dynamic Memory allocation.....15

EXPERIMENT-10
Implement programs using Link List concept. 16

EXPERIMENT-11
Implement the program for Pre-Processor. 17

LABWORK BEYOND CURRICULA

EXPERIMENT 12.....18
- WAP to implement Stack using Linked Lists.
- WAP to implement Queue using Linked Lists.

EXPERIMENT 13…….. 18
- WAP to implement Single Linked List.
- WAP to implement Doubly-Linked List.

LIST OF EXPERIMENTS

1 Arrays
2 Character Arrays & Strings
3 User-defined Functions
4 Recursion & Arrays as arguments to function
5 Structures
6 Unions
7 Pointers
8 Files & Command line arguments
9 Dynamic Memory allocation
10 Linked List
11 The Pre-Processor

COMMON PROCEDURE

Tools / Apparatus: Unix/Linux/Windows Operating System, Text Editor, gcc compiler or Turbo C++ IDE

Procedure:

- Prepare the Box Diagram of the Program
- Prepare the Flow Chart of the Program
- Write the code of the program
- Compile the program for any compile-time errors
- Run the program
- Debug the program for any errors

LABWORK BEYOND CURRICULA

Advanced Linked Lists

- WAP to implement Stack using Linked Lists.
- WAP to implement Queue using Linked Lists.
- WAP to implement Single Linked List.
- WAP to implement Doubly-Linked List.

Sample Experiment

1 AIM: Define a structure for items. The members are item number, item name, item price. Take all the details for at least 5 items. Using function search for the particular item by its name or by its number.

2 TOOLS/APPARATUS: Turbo C or gcc compiler

3 STANDARD PROCEDURES:

COMMON PROCEDURE:

- Step 1: Create a folder in either E or F drive with your Id Number or Name Followed by RollNo.
- Step 2: Start the TC (Turbo C) from Desktop Icon or Go To Directory D:/TC/BIN/ and run tc.exe . An Editor will be start.
- Step 3: Click on File Menu --> New. New (.c) file will be created. Again Click on File -> Save an dialog box is going open write the path to your directory
  e.g. E:\structure\FileName.C and Press OK. Now your C program is going to save at your directory.
- Step 4: Go To Option->Directories Check That Include Directory is Set As D:\TC\Include and Library Directory is Set To D:\TC\LIB

3.1 Analyzing the Problem:

- First create a file named as “item.c” .
- After that includes the standard input/output files.
- Now define the structure “item” and its members.
- Create the functions which are necessary for the program.
- Now create the main function and take the information.

3.2 Designing the Solution:

- Create a c file name as “item.c”.
- Define the structure named as “item”. Also define its members that are item number, item name, item price.
- Make a function to search a record by an item number that is “searchitembyno()” or by item name that is “searchitembyname()”.
- Now in the main function declare a variable of the structure item “it[5]”. It should be an array because we want all the details for at least 5 items.
- Now take the all the details by the user using scanf() function.

- Now enter the choice by which you want to search. And call the function for desire output. And display the details using printf() function.

3.3 Implementing the Solution:

3.3.1 Writing Source Code:

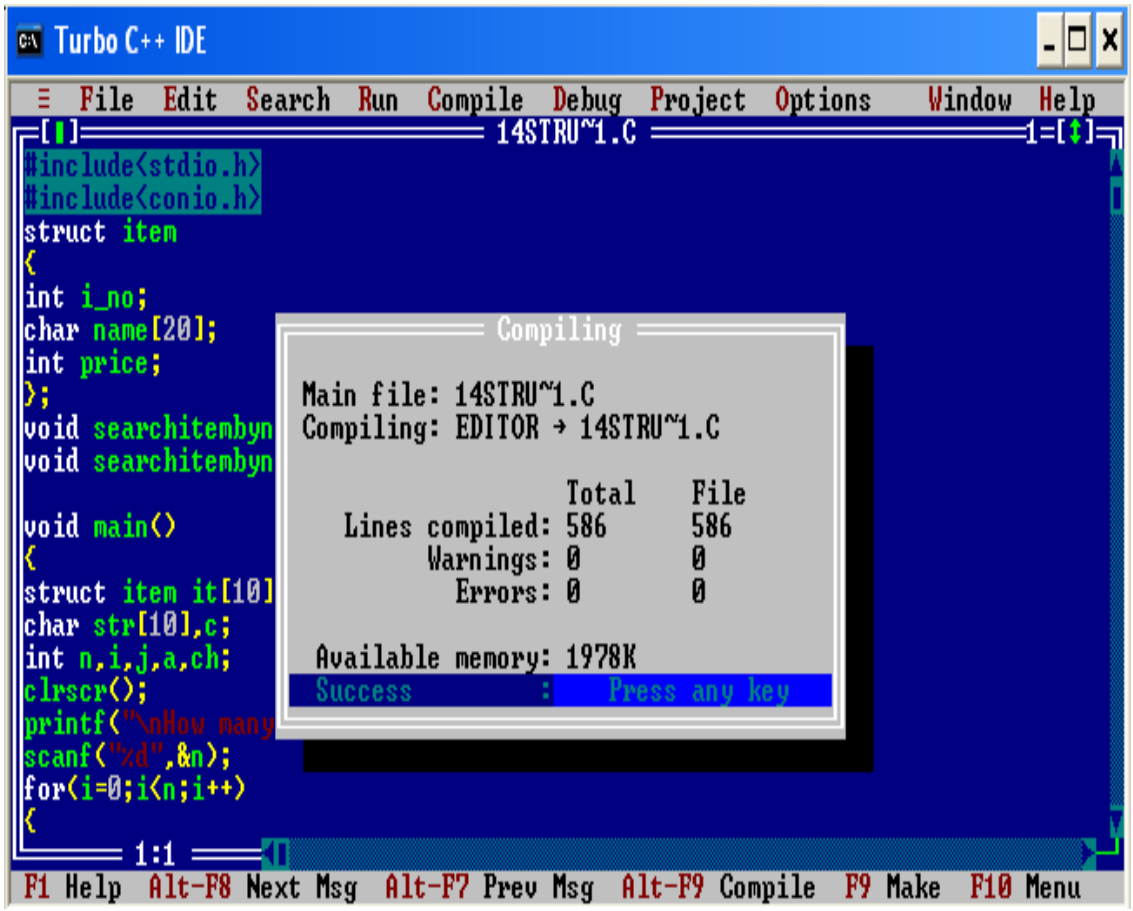
```
#include<stdio.h>
#include<conio.h>

struct item                 //defining the structure name item
{
int no;
char name[20];
int price;
};

void searchitembyno(struct item i1[ ] ,int,int);  //function prototype to search by item number

void searchitembyname(struct item i1[ ],char [ ],int); //function prototype to search by item name

void main( )
{
  struct item it[10],t;
  char str[10],c;
  int no,i,j,a,ch;
  clrscr();
  printf("\nHow many item :");
  scanf("%d",&n);              // take n number of items
  for(i=0;i<n;i++)
   {
      printf("\nEnter no, name and price of item :");
      scanf("\n%d %s %d",&it[i].i_no,it[i].name,&it[i].price);
 }

label:

// now for searching there are two options
printf("\n\nEnter 1 to search by number and 2 to search by name : ");
scanf("%d",&ch);

switch(ch)
{
case 1:   printf("\n\nEnter the no for the item u want to search : ");
          scanf("\n%d",&no);
          printf("\n");
          searchitembyno(it,a,no); //this is a function call to search by number
          break;
```

```
case 2:    printf("\n\nEnter the name of the item u want to search : ");
           scanf("\n%s",str);
           printf("\n");
           searchitembyname(it,str,n); //this is a function call to search by name
           break;
default:   break;
}

printf("\n want to search again ? y/n ");  //to search again Enter 'y' or to stop searching Enter 'n'.

c=getche();
if(c=='y')
{
 goto label ;
}

  getch();
}
//function definition ro search by number
void searchitembyno(struct item i1[],int a,int n)
 {
     int i,flag=0;
     for(i=0;i<n;i++)
     {
         if(i1[i].no==a)      //comparing nos
          {
            flag=1;
            printf("\n name and price is   %s  %d :",i1[i].name,i1[i].price);
            break;
          }
    }

   if(flag==0)
    {
      printf("\nitem not found ");
    }
}

//function definition ro search by name
void searchitembyname(struct item i1[],char s[],int n)
  {
     int i,flag=0;
     for(i=0;i<n;i++)
      {
          if((strcmp(i1[i].name,s))==0)    //comparing string
           {
```

```
            flag=1;
            printf("\n no name and price is %d  %s  %d :",i1[i].no,i1[i].name,i1[i].price);
            break;
          }
     }

    if(flag==0)
     {
       printf("\nitem not found ");
    }
}
```

3.3.2 Compilation /Running and Debugging the Solution:

- Go to Compile Menu and Press the Compile For the Compilation of the code.



- If Successful Compilation is done then Run the Code Using ctrl + F9 key.

```
How many item :5

Enter no, name and price of item :1
munch
10

Enter no, name and price of item :2
colgate
15

Enter no, name and price of item :3
pepsi
20

Enter no, name and price of item :4
rice
50

Enter no, name and price of item :5
notebook
15


Enter 1 to search by no and 2 to search by name : 1


Enter the no for the item u want to search : 4


 name and price is   rice  50 :
 want to search again ? y/n y

Enter 1 to search by no and 2 to search by name : 2


Enter the name of the item u want to search : pepsi


 no name and price is 3  pepsi  20 :
 want to search again ? y/n n
```

3.4 Testing the Solution:

- User must have entered all the details with respected to its data type.
- In search by name or number if that record is found than it will display the desire output. Otherwise it will display that item not found.
- If we enter a character instead of integer than the output will be unpredictable.
- Same way if we enter an integer instead of character than the output will be unpredictable.

4 Conclusions:

Hence we have concluded that this experiment will give us the knowledge that how to code a meaningful and understandable program, as well as how to analyze, design and test the program.

Required Software/ Software Tool:

-Linux Operating System and/ or Windows Operating System
-Turbo C/C++ IDE.

Common procedure:

Step 1**:** For the given problem statement design
Flowchart/Algorithm/Logic
Step 2: Define variables and functions which will show the flow of program
Step 3: Write C code in the file with .c extension
Step 4: If the OS is Windows than compile and run .c file using turbo C.
Compile code using gcc compiler for Linux, which will create a.out executable file.
Step 5: Test the program using sample input and write down output.

EXPERIMENT-1

Aim: Implement the programs of Arrays.

1) Find average of N numbers using arrays.
2) Addition of two 3X3 matrices.
3) Multiplication of two 3X3 matrices
4) Sorting of N numbers
5) Inserting an element in an array

Tools: Turbo C or gcc compiler.

Procedure:

1) Create an array for n numbers where the value for n is given by user. Using 1-D array you can take the value and using loop you can find the sum as well as average for n numbers.
2) Using 2-D arrays and loops you can generate the 3x3 matrix. After that using loop add the two matrices.

```
c[i][j]=a[i][j]+b[i][j];
```

3) Same way multiply the two 3x3 matrices based on multiplication rule in matrix.

```
c[i][j]+=a[i][k]*b[k][j]; // using 3 loops
```

4) Apply the logic of sorting for n element in an array. Using two loops compare each and every element.

```
if(a[j]>a[j+1])
 {
        temp = a[j];
        a[j]=a[j+1];
        a[j+1]=temp;
 }
```

5) Take the appropriate position or index where you want to insert a new element. After that shift other element down, create the space for new element and insert it.

EXPERIMENT-2

Aim: Implement the programs of Character Array.

1) WAP that will read a string and display its length as output (without using strlen() function).
2) WAP that will read two strings.
   If given two strings are same then print "Given strings are same".
   If given two strings are not same then print "Given strings are not same"
3) WAP that will concatenate one string to the end of another string.
4) WAP which searches for the pattern in the subject string and returns the position/index of the start of the string where match is found.

Tools: Turbo C or gcc compiler.

Procedure:

1) Using loop take the string from the user. And just increment the value of the counter by 1 for each and every character. And find the length of the string.
2) Take two strings from the user using loop, after that compare each and every character of one string with another string and display appropriate message.

```
for(i=0;i<=len1;++i)
{
        if(s1[i]!=s2[i])
        {
                flag=1;
                break;
        }
}
```

3) Take two strings from the user. Size of first string should be larger enough to store both the strings.
   Using loop reach at the end of first string, then take the elements of other string one by one and append it.

```
for(j=0,i=len1;j<=len2;++i,j++)
{
        s1[i]=s2[j];
}
```

4) Take two strings from the user. Consider second string as a pattern and write logic which search a pattern in a subject string and display appropriate message.

EXPERIMENT-3

Aim: Write following programs using Functions.

1) Printline() which prints '=' sign 81 times in the same line.
2) Write a function to calculate and display the total amount given that
   Total_amount=p*pow((1+r),n)
   Where p = principle amount
   r= rate of interest
   n = period.
3) Modify program (above program) for returning total amount
4) Implement Fibonacci of n series using function (Iterative version)

Tools: Turbo C or gcc compiler.

Procedure:

1) Make a function which prints the '=' sign 81 times using a loop.
2) Make functions which calculate the total amount by the formula which is given.
3) Modify the above program which returns the total amount.
4) Make an iterative version of Fibonacci series using function.

```
next = first + second;          // in a loop
first = second;
second = next;
printf("%d\n",next);
```

EXPERIMENT-4

Aim: Implement programs using Recursive function & Arrays as arguments to function.

1) Factorial of n using recursive (non-iterative) function.
2) Sorting of numbers in ascending order using function with array as arguments.
   -Bubble sort
   -Selection sort
   -Insertion sort

Tools: Turbo C or gcc compiler.

Procedure:

1) Make a recursive function for factorial.
   f=n*fact(n-1); //fact is a function
2) Make a program which sort the given numbers using function in which array is given as an argument.

-Bubble sort

Repeatedly swap the adjacent elements if they are in wrong order.

```
if (arr[j] > arr[j+1])
   swap(&arr[j], &arr[j+1]);
```

-Selection sort

- Selection Sort selects the kth smallest element in a[1…n] and places it in the k^{th} position of a[ ].
- The remaining elements are rearranged such that a[m]<=a[k] for 1<=m<k and a[m]>=a[k] for k<m<=n.

-Insertion sort

- Function insert takes as a parameter an array containing a sequence of length n that is to be sorted.
- The input array A contains the sorted output sequence when insert function is finished.

EXPERIMENT-5

Aim: Implement the programs using Structures.

1) Generate a result table which consists of student id, student name, marks of three subject and total marks. Write a program which takes input for ten students and displays result table. Also display student information separately who got the highest total.
2) Create a structure containing information of products viz, id, name, price etc. Implement search function on basis of id of product and displays its respective name and price.
3) Modify above program and sort product details according to name(alphabetically order)

Tools: Turbo C or gcc compiler.

Procedure:

1) Design a structure with members as student id, student name, marks of three subject and total marks.
   -Using loop take information for ten students and displays result table.
   -Apply logic to get highest total marks and display all information.
2) Create a structure for product with members as id, name, and price.
   -Take id as input and search a product from the list of product and display all information.
3) Modify the above program, apply the logic of sorting and swap the whole structure and display all the details.

EXPERIMENT-6

Aim: Implement the programs using Union.

1) Read and display the members of Union (Take book details-ISBN no , Title, Price)
2) Store information of 10 persons. Information includes name and age. But criteria is for the child age should be in form of full birth date, for an adult the age should be in years only, while for aged person store age indicating the status 'O'. Use union for memory efficiency.

Tools: Turbo C or gcc compiler.

Procedure:

1) Design Union with members as book ISBN no, titale and price.
   -Take the information from user and display it.
   -Analyze the output.
2) Design Union for person details. Take the input for 10 people and display the information as described.

EXPERIMENT-7

Aim: Implement the programs using Pointers.

1) Swap/exchange values of two integer variables using function. Use pointers.
2) Write a function mystrcat(s,t) which copies string t to the end of the string s.
3) Addition of two 3X3 matrices using pointers.

Tools: Turbo C or gcc compiler.

Procedure:

1) Exchange value of two variable using pass by address method.
   - Create a separate function swap() with two integer pointers as arguments.

```
void swap (int *a1 , int * b1)
{    temp=*a1;
     *a1=*b1;
     *b1=temp;
}
```

2) Create mystrcat(char *s ,char *t) function, append the string t at the end of string s.
3) Create two 3X3 array, using pointer add the corresponding elements in array.

```
*(*(c+i)+j)=*(*(a+i)+j)+ *(*(b+i)+j); // a[3][3],b[3][3],c[3][3]
```

EXPERIMENT-8

Aim: Implement programs for Files & Command line arguments.

1) Read input characters from user and write it to file. After that read the content of the same file and display on screen.
2) Implement copy command with the use of command line arguments. Like mycopy source.txt destination.txt copies content of source file to destination file.
3) Write integer numbers to a file. Read them back and find average of all numbers.

Tools: Turbo C or gcc compiler.

Procedure:

1) Take the input from the user and using functions available for the file write the data to the file. After that again read the data from the file and displays it on the screen.
2) Use the command line argument to read the argument given by the user and using that make the program which copy the content of one file to the another file.
3) Make a program which takes the input from the user for integer values write this values to a file. Again read it back and find the average.

EXPERIMENT-9

Aim: Implement programs for Dynamic Memory allocation.

1) WAP that uses a table of integers whose size will be specified interactively at runtime.
2) WAP to store character string in a block of memory space created by malloc and then modify the same to store a larger string.

Tools: Turbo C or gcc compiler.

Procedure:

1) Create a list of integer using calloc, take the size as input from the user.
2) Create a list of characters using malloc. Take the size from the user. Using realloc modify the size of string and display it.

EXPERIMENT-10

Aim: Implement programs using Link List concept.

1) WAP to create a link list. Allocate the memory at run time whenever required using malloc.
2) WAP to insert a node in a link list.
3) WAP to delete a particular node in a link list.

Tools: Turbo C or gcc compiler.

Procedure:

1) Create a list in following method:
   - Step 1 Read the element into x
   - Step 2 Create a temp node in the memory
     temp =(struct node)sizeof (node)
   - Step 3 Assign the values in temp node as follows
     temp -> info =x
     temp ->next=null
   - Step 4 check whether head is null or not
     if (head=null) { head=temp ; current=temp }
     else { current ->next =temp ; current ->current ->next }
   - Step 5 follow steps 1 to 4 to insert remaining element in the list.

2) Insert the node in following way:
   - Step 1 Read the element into x
   - Step 2 Create an temp node in memory as follows
     temp=(struct node *)size of (node)
   - Step 3 Set the values in temp node as follows
     temp-> info =x
     temp->next=null
   - Step 4 Search the element after which node will be inserted
     current =SEARCH()
   - Step 5 insert temp node after current node as follows
     temp->next =current -> next
     current->next=temp

3) Similarly apply the logic to delete a node in a link list. Search an element in the list and accordingly change the position of the next pointer.

EXPERIMENT-11

Aim: Implement the program for Pre-Processor.

1) Define a macro print_value that can be used to print two values of arbitrary type.
2) Compute Area of circle using three ways of macro substitution

Tools: Turbo C or gcc compiler.

Procedure:

1) To define a macro print_value many ways are there. One of the way to define is as follow:

   #define print_value(x,y) printf("%d %d ", x,y)
2) Use the three different ways for macro substitution 1) simple macro 2) macro with argument 3) Nesting of macros. Using all these three method compute the area of circle.

LABWORK BEYOND CURRICULA

EXPERIMENT 12:

- WAP to implement Stack using Linked Lists.
- WAP to implement Queue using Linked Lists.

EXPERIMENT 13:

- WAP to implement Single Linked List.
- WAP to implement Doubly-Linked List.