C Programming
(Assignment –I)

Submitted in partial fulfilment of the requirements for the degree of

Post Graduate Diploma in Information Technology

by

Vijayananda D Mohire

(Registration No.200508208)

Information Technology Department
Symbiosis Bhavan,
1065 B, Gokhale Cross
Road, Model Colony, Pune – 411016,
Maharashtra, India
(2007)

C Programming

Table of Contents

Question 1 6
Answer 1(a) 6
Answer 1(b) 6
Question 2 7
Answer 2 7
Question 3 10
Answer 3 10
Question 4 11
Answer 4 11
Question 5 12
Answer 5 12
Question 6 13
Answer 6 13
Question 7 15
Answer 7 15
Question 8 16
Answer 8 16
Question 9 18
Answer 9 18
Question 10 19
Answer 10 19
Question 11 20
Answer 11 20

C Programming

Question 1 Write algorithm for the following:

a) To check whether an entered number is odd / even.

b) To calculate sum of three numbers.

Answer 1(a)

Code:

```
# include <stdio.h>
 main (void)
 {
 int input; /* define variable to store user input */
 int mod; /* define variable to store modulus 2 output */

 clrscr(); /* Clear screen*/

 do

        {
    printf ("\n Please enter a Number(Enter Zero to Quit): \n");
    scanf("%d", &input);  /* Get the formatted Input number */

    mod = input%2;  /* Modulus 2 operation that returns 0 if number is even.*/

    if (mod == 0)  /* Check if the return value and output as Even or Odd */
    printf ("%s", "Number is Even\n");

          else
          printf("%s", "Number is odd\n");

      } while (input != 0);
  }
```

Answer 1(b)

Code:

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

 main(void)
```

```
  {

int i,j,k,sum; /* define variable to store user inputs and sum */

clrscr(); /* Clear screen*/

printf (" Enter Number 1:\n");
scanf ("%d",&i); /* Get the formatted Input number 1 */

printf ("\n Enter Number 2:\n");
scanf ("%d",&j); /* Get the formatted Input number 2 */

printf ("\n Enter Number 3:\n");
scanf ("%d",&k); /* Get the formatted Input number 3 */

sum = i+j+k; /* Calculate the Sum of the 3 inputs */

printf ("Sum of 3 Numbers is:\n%d",sum); /* Prints the calculated sum */
getch(); /* Pause so that results are visible */
  }
```

HINT: You can put this in a do while loop and check an increment counter

Evaluator's Comments if any:

Question 2 Write short notes on the following:

a) C Variables

Answer 2

Like most programming languages, C is able to use and process named variables and their contents.

Variables are most simply described as names by which we refer to some location in memory - a location that holds a value with which we are working. It often helps to think of variables as a "pigeonhole", or a placeholder for a value. You can think of a variable as being equivalent to its value. So, if you have a variable **i** that is initialized to 4, i+1 will equal 5.

All variables in C are typed. That is, you must give a type for every variable you declare.

C data types

In Standard C there are four basic data types. They are int, char, float, and double.

The int type

The int type stores integers in the form of "whole numbers". An integer is typically the size of one machine word, which on most modern home PCs is 32 bits (4 octets). Examples of literals are whole numbers (integers) such as 1, 2, 3, 10, 100... When int is 32 bits (4 octets), it can store any whole number (integer) between -2147483648 and 2147483647. A 32 bit word (number) has the possibility of representing 4294967296 numbers (2 to the power of 32).

If you want to declare a new int variable, use the int keyword. For example:

```
    int numberOfStudents, i, j=5;
```

In this declaration we declare 3 variables, numberOfStudents, i & j, j here is assigned the literal 5.

The char type

The char type is similar to the int type, yet it is only big enough to hold one ASCII character. It stores the same kind of data as an int (i.e. integers), but always has a size of one byte. It is most often used to store character data, hence its name.

Examples of character literals are 'a', 'b', '1', etc., as well as special characters such as '\0' (the null character) and '\n' (endline, recall "Hello, World").

When we initialize a character variable, we can do it two ways. One is preferred, the other way is bad programming practice.

The first way is to write : char letter1='a';

This is good programming practice in that it allows a person reading your code to understand that letter is being initialized with the letter "a" to start off with.

The second way, which should not be used when you are coding letter characters, is to write : char letter2=97; /* in ASCII, 97 = 'a' */

This is considered by some to be extremely bad practice, if we are using it to store a character, not a small number, in that if someone reads your code, most readers are forced to look up what character corresponds with the number 97 in the encoding scheme.

There is one more kind of literal that needs to be explained in connection with chars: the string literal. A string is a series of characters, usually intended to be output to the string. They are surrounded by double quotes (" ", not ' '). An example of a string literal is the "Hello, world!\n" in the "Hello, World" example.

The float type
float is short for Floating Point. It stores real numbers also, but is only one machine word in size. Therefore, it is used when less precision than a double provides is required. float literals must be suffixed with F or f, otherwise they will be interpreted as doubles. Examples are: 3.1415926f, 4.0f, 6.022e+23f. float variables can be declared using the float keyword.

The double type
The double and float types are very similar. The float type allows you to store single-precision floating point numbers, while the double keyword allows you to store double-precision floating point numbers - real numbers, in other words, both integer and non-integer values. Its size is typically two machine words, or 8 bytes on most machines. Examples of double literals are 3.1415926535897932, 4.0, 6.022e+23 (scientific notation). If you use 4 instead of 4.0, the 4 will be interpreted as an int.

Evaluator's Comments if any:

Question 3 Accept principal amount, rate of interest, and duration from the user. Display Interest Amount and Total Amount (Principal + Interest).

Answer 3

```
   Code:
   #include <stdio.h>
   main (void)
   {

   /* define variable to store Principal, Percentage Rate of Interest,Time, Interest
amount and Total amount */
   float PrincipalAmt, PercRateofInt, RateofInt, Time, IntAmt, TotalAmt;

   clrscr(); /* Clear screen*/

   printf("Enter Principal Amount:\n");
   scanf("%f", &PrincipalAmt); /* Get the formatted Principal Amount */

   printf("Enter Percentage Rate of Interest:\n");
   scanf("%f", &PercRateofInt); /* Get the formatted Percentage rate of Interest*/

   printf("Enter the Duration in years:\n");
   scanf("%f", &Time);/* Get the formatted Time period in years */

   RateofInt = PercRateofInt/100; /* Convert Percentage to float equivalent for Rate
of Interest*/

   IntAmt = PrincipalAmt*RateofInt*Time; /* Compute Interest Amount using P*R*T
formula */

   TotalAmt = PrincipalAmt+IntAmt; /* Compute Total Amount using P+I formula */

   printf("Interest Amt is:\n%f\n",IntAmt); /* Prints the computed Interest */
   printf("Total Amt is:\n%f",TotalAmt); /* Prints the computed Total amount */

   getch(); /* Pause so that results are visible */
      }
```

Question 4 Accept any number from the user. Display whether the number is divisible by 100 or not.

Answer 4
Code:

```
 main (void)

  {

  int Num, Remainder; /* define variable to store Number and Remainder */

  clrscr(); /* Clear screen*/

  printf ("Enter a Number:\n");
  scanf ("%d",&Num); /* Get the formatted Number */

  Remainder = Num % 100; /* Divide the Number by 100 and get the remainder */

  if ( Remainder == 0 ) /* check if the Remainder is equal to zero or no, so that the
divisibility can be printed */
  printf("Entered Number is divisible by 100\n");
      else
          printf("Entered Number is not divisible by 100\n");
          getch(); /* Pause so that results are visible */

 }
```

Evaluator's Comments if any:

Question 5 Write a program to swap the values of two numbers. Do this using call by reference method of function.

Answer 5
Code:

```
    #include < stdio.h > void swap(int * firstnum, int * secondnum); /* function
prototype for call by ref*/

     main(void) {

        int firstnum, secondnum; /* define variables to store entered numbers */

        clrscr(); /* Clear screen*/


        printf("Please enter first Number:");

        scanf("%d", & firstnum); /* Get the formatted firstnumber */


        printf("Please enter second number:");

        scanf("%d", & secondnum); /* Get the formatted secondnumber */


        printf("\nBefore Swamp, firstnum is: %d\t, secondnum is: %d\n", firstnum,
secondnum);

        /* Pass the numbers to be swapped, pass by reference */

        swap( & firstnum, & secondnum);

        printf("After swap, firstnum is: %d\t, secondnum is: %d\n", firstnum,
secondnum);

        getch(); /* Pause so that results are visible */

     }
```

```
void swap(int * first, int * second) {
   int temp; /* define variables for temporary storage */
   temp = * second; /* Assign the value of second to temp variable */
   * second = * first; /* Assign the value of the first to second variable */
   * first = temp; /* Assign the value of the temporary variable to first variable */
}
```

Question 6 Accept a month in digit from the user. Display the month in words. If number is not between 1 and 12 display message “Invalid Month”. (Use ‘switch’)

Answer 6
Code:

```
#include <stdio.h>
main()
{

int month; /* define variable to store entered month */
clrscr(); /* Clear screen*/

printf("Enter the Month in range 1-12:\n");
scanf("%d",&month); /* Get the formatted month */

if (month < 1) /* Signal error is entered integer is not within range 1 to 12 */
printf("Invalid Month, please enter in valid range");
if (month > 12)
printf("Invalid Month, please enter in valid range");
printf("\n");

switch (month) /* Use the switch case to control what needs to be printed  */
   {
   case 1:
   printf("You entered January");
   break;
```

```
case 2:
printf("You entered February");
break;

case 3:
printf("You entered March");
break;

case 4:
printf("You entered April");
break;

case 5:
printf("You entered May");
break;

case 6:
printf("You entered June");
break;

case 7:
printf("You entered July");
break;

case 8:
printf("You entered August");
break;

case 9:
printf("You entered September");
break;

case 10:
printf("You entered October");
break;

case 11:
printf("You entered November");
break;

case 12:
printf(" You entered December");
break;

default:
break;
```

```
    }
    getch(); /* Pause so that results are visible */

}
```

Question 7 Accept any two numbers from the user. Using pointers swap the values two numbers without using third variable

Answer 7

```
 #include<stdio.h>
  void swap(int *firstnum, int *secondnum);/* function prototype for call by ref*/
  main()

  {
  int firstnum, secondnum; /* define variables to store entered numbers */
  clrscr();/* Clear screen*/

  printf("Please enter first Number:");
  scanf("%d",&firstnum); /* Get the formatted firstnumber */

  printf("Please enter second number:");
  scanf("%d",&secondnum); /* Get the formatted secondnumber */

  printf("\nBefore Swamp, firstnum is: %d\t, secondnum is: %d\n",
firstnum,secondnum);

  swap(&firstnum,&secondnum); /* Pass the numbers to be swapped, pass by
reference */

  printf("After swap, firstnum is: %d\t, secondnum is: %d\n",firstnum,secondnum);
  getch(); /* Pause for results */
  }

  void swap(int *firstnum, int *secondnum)
```

```
{
   if( firstnum!=secondnum)
      {

      /* Use the ^ operator to swap without temp variable*/
      *firstnum ^= *secondnum;
      *secondnum ^= *firstnum;
      *firstnum ^=  *secondnum;
}          }
```

Question 8 Create a structure to store the employee number, name, department and basic salary. Create an array of structure to accept and display the values of 10 employees

Answer 8

Code:

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

void main()
{
    struct emp    /* Structure declaration */
    {
        int empnum;
        int basicsal;
        char name[20];
        char dept[20];
     } ;

struct emp My_emps[10]; /* Structure array declaration  */
int hcount = 0;         /* Count of the number of EMPS */
int i = 0;              /* Loop counter  */
char test = '\0';       /* Test value for ending */
```

```
    clrscr();

    for(hcount = 0; hcount < 10 ; hcount++ )
    {

    printf("\nDo you want to enter details of a%s employee (Y or N)? ",
                                        hcount?"nother " : "" );
        scanf(" %c", &test );
        if(tolower(test) == 'n')
          break;

    printf("\nEnter the name of the employee: " );
    scanf("%s", My_emps[hcount].name );   /* Read the emp's name */

    printf("\nEnter employee number: " );
    scanf("%d", &My_emps[hcount].empnum );   /* Read the emp's num  */

    printf("\n Enter dept of employee: "  );
    scanf("%s", My_emps[hcount].dept );

    printf("\nEnter basic salary: ");
    scanf("%d",&My_emps[hcount].basicsal );

      }

      /* Now display the employee details */
      for (i = 0 ; i < hcount ; i++ )
          {
        printf("\n\nEmp%d: %s\t%d\t%s\t%d",
        i,My_emps[i].name, My_emps[i].empnum,
My_emps[i].dept,My_emps[i].basicsal);
           }

    getch(); / * Pause for viewing results */

    }
```

Question 9 Accept a file name from the user. Display the contents of the file. Also add the entered string to the file.

Answer 9

Code:

```
#include <stdio.h>
   #include <io.h>
   #include <stdlib.h>

   int main()
   {
   FILE *fp;        /* define file pointer variable */
   char fname[100];
   char s[100];
   int t;

   clrscr();/ Clear screen*/

   printf("Enter filename( Eg: Vijay.txt\n");
   scanf("%s",fname); /* Get the formatted filename */

   printf("%s",fname);
   if((fp = fopen(fname,"a")) == NULL)
      {
         printf("cannot open file.\n");
         exit(1);/* Exit if there is error opening file */
      }

   printf("\nEnter a string:"); /* Get the entered string */
   fscanf(stdin,"%s",s);
   fprintf(fp,"%s\n",s); /* Append the value to the file */
   fclose(fp);

   if((fp = fopen(fname,"r"))== NULL)
      {
         printf("cannot open file.\n");
         exit(1); /* Exot if error opening file */
      }
   fscanf(fp,"%s",s);
   /* Display the contents*/
   fprintf(stdout," File created in current dir and the updated content: %s",s);
   getch();
   return 0;
   }
```

Question 10 Accept any number as a command line argument. Write a program to display the number in reverse order.

Instructions: Please run this from Command prompt to see correct results.

Answer 10

Code:

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

main (int argc,char *argv[])
{

int p=1,c=0,i, j;  /* define variable  and let p point to arg 1*/

char copy[10];

clrscr(); /* Clear screen*/

printf("Entered Number is: %s\n", argv[p]);

strcpy(copy,argv[p]);  /* Copy the command line argv[1] data to local variable  */
printf("copy %s\n", copy);

for(i=0,j=strlen(copy)-1;i <j;i++,j--) /* reverse the chars from the local variable */
   {
       c =copy[i];
       copy[i] = copy[j];
```

```
        copy[j] =c;
     }
  printf("copy after rev %s\n",copy);    /* print out the reversed string */
  getch(); /* pause to view results*/
 }
```

Question 11 Write a short note on enum

Answer 11

enum is the abbreviation for ENUMERATE, and we can use this keyword to declare and initialize a sequence of integer constants. Here's an example:

```
enum colors {RED, YELLOW, GREEN, BLUE};
```

Here, colors is the name given to the set of constants. Now, if you don't assign a value to a constant, the default value for the first one in the list - RED in our case, has the value of 0. The rest of the undefined constants have a value 1 more than the one before, so in our case, YELLOW is 1, GREEN is 2 and BLUE is 3.

But you can assign values if you wanted to:

```
enum colors { RED=1, YELLOW, GREEN=6, BLUE };
```

Now RED=1, YELLOW=2, GREEN=6 and BLUE=7.

The main advantage of enum is that if you don't initialize your constants, each one would have a unique value. The first would be zero and the rest would then count upwards.