

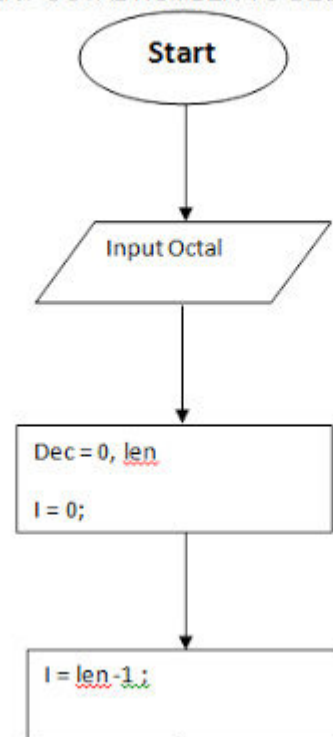
MCS-011 Solved assignment july 2017-january 2018 session-Sunil Poonia

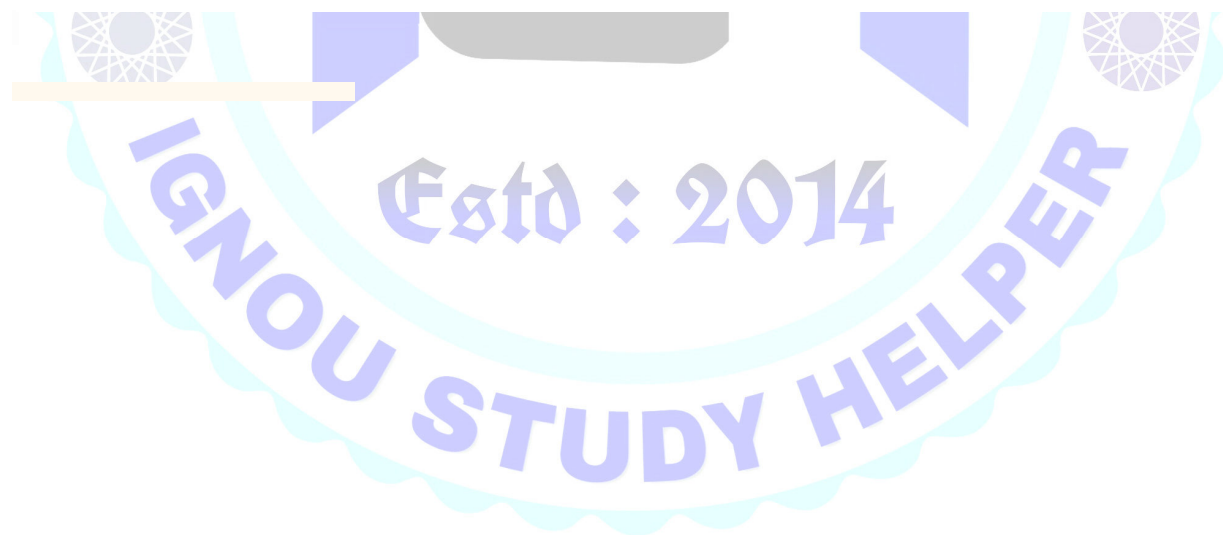
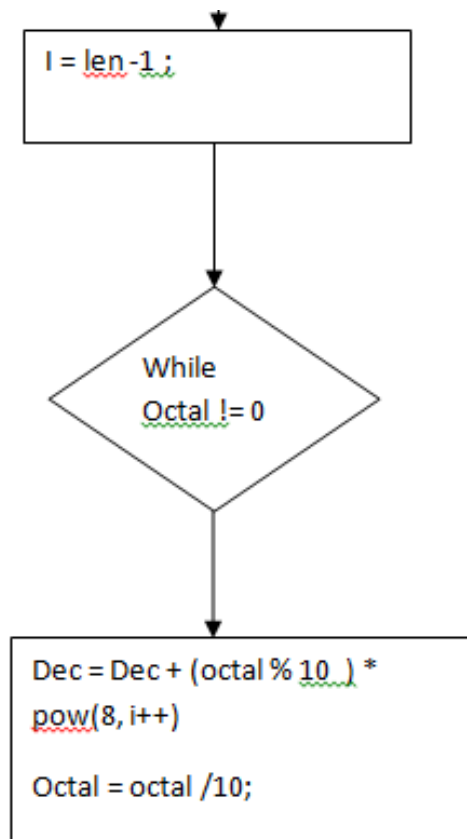
1)

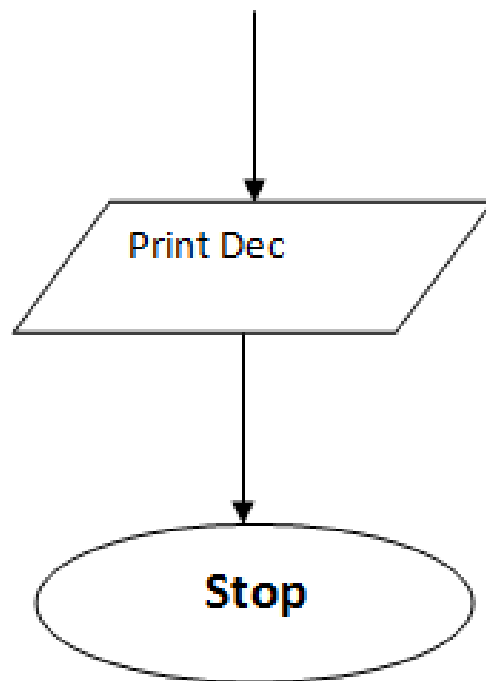
Octal is a numbering system that uses eight digits, 0 to 7, arranged in a series of columns to represent all numerical quantities. Each column or place value has a weighted value of 1, 8, 64, 512, and so on, ranging from right to left. Decimal is a term that describes the base-10 number system commonly used by lay people in the developed world.

FLOWCHART TO CONVERT OCTAL NUMBER TO DECIMAL:

FLOWCHART TO CONVERT OCTAL NUMBER TO DECIMAL:







C PROGRAM TO CONVERT OCTAL NUMBER TO DECIMAL:

```
/*  
 * C Program to Convert Octal to Decimal  
 */  
#include <stdio.h>  
#include <math.h>  
  
int main()  
{
```

```

long int octal, decimal = 0;
int i = 0;

printf("Enter any octal number: ");
scanf("%ld", &octal);
while (octal != 0)
{
    decimal = decimal +(octal % 10)* pow(8, i++);
    octal = octal / 10;
}
printf("Equivalent decimal value: %ld",decimal);
return 0;
}

```

Output:

Enter any octal number: 67
Equivalent decimal value: 55

2)

This C Program performs ATM transaction. The types of ATM transaction are

- 1) Balance checking
- 2) Cash withdrawal
- 3) Cash deposition.

You can opt any of the above transaction according to your need of transaction.

ALGORITHM TO ILLUSTRATE AN ATM MONEY WITHDRAWL OPERATION FROM USER'S SAVING ACCOUNT.

1. acc balance starts at zero
2. do not use global variables
3. loop the main menu after every transaction until user presses exit
4. after every transaction user returns to the main menu.

menu

- a. deposit
- b. withdraw
- c. balance inquiry
- d. fast cash - a-100
b-200
c-500
d-1000

and exit

C PROGRAM TO ILLUSTRATE AN ATM MONEY WITHDRAWL OPERATION FROM USER'S SAVING ACCOUNT.

*

* C Program to Display the ATM Transaction

*/

#include <stdio.h>

```
unsigned long amount=1000, deposit, withdraw;
int choice, pin, k;
char transaction ='y';
```

```
void main()
{
    while (pin != 1520)
    {
        printf("ENTER YOUR SECRET PIN NUMBER:");
        scanf("%d", &pin);
        if (pin != 1520)
            printf("PLEASE ENTER VALID PASSWORD\n");
    }
    do
    {
        printf("*****Welcome to ATM Service*****\n");
        printf("1. Check Balance\n");
        printf("2. Withdraw Cash\n");
        printf("3. Deposit Cash\n");
        printf("4. Quit\n");
        printf("*****?*****?\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
            case 1:
                printf("\n YOUR BALANCE IN Rs : %lu ", amount);
                break;
            case 2:
                printf("\n ENTER THE AMOUNT TO WITHDRAW: ");
                scanf("%lu", &withdraw);
                if (withdraw % 100 != 0)
                {
                    printf("\n PLEASE ENTER THE AMOUNT IN MULTIPLES OF 100");
                }
                else if (withdraw >(amount - 500))
                {
                    printf("\n INSUFFICIENT BALANCE");
                }
                else
                {
                    amount = amount - withdraw;
                    printf("\n\n PLEASE COLLECT CASH");
                    printf("\n YOUR CURRENT BALANCE IS%lu", amount);
                }
                break;
            case 3:
                printf("\n ENTER THE AMOUNT TO DEPOSIT");
```

```

        scanf("%lu", &deposit);
        amount = amount + deposit;
        printf("YOUR BALANCE IS %lu", amount);
        break;
    case 4:
        printf("\n THANK U USING ATM");
        break;
    default:
        printf("\n INVALID CHOICE");
    }
    printf("\n\n DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n): \n");
    fflush(stdin);
    scanf("%c", &transaction);
    if (transaction == 'n' || transaction == 'N')
        k = 1;
    } while (!k);
    printf("\n\n THANKS FOR USING OUT ATM SERVICE");
}

```

Output:

ENTER YOUR SECRET PIN NUMBER:1520

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: 1

YOUR BALANCE IN Rs : 1000

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: 2

ENTER THE AMOUNT TO WITHDRAW: 200

PLEASE COLLECT CASH

YOUR CURRENT BALANCE IS800

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: 3

ENTER THE AMOUNT TO DEPOSIT5000

YOUR BALANCE IS 5800

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: 1

YOUR BALANCE IN Rs : 5800

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: 4

THANK U USING ATM

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: 4

THANK U USING ATM

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

*****Welcome to ATM Service*****

1. Check Balance
2. Withdraw Cash
3. Deposit Cash
4. Quit

*****?*****?

Enter your choice: n

THANK U USING ATM

DO U WISH TO HAVE ANOTHER TRANSCATION?(y/n):

THANKS FOR USING OUT ATM SERVICE.

3)

Recursion is a programming technique that allows the programmer to express operations in terms of themselves. In C, this takes the form of a function that calls itself.

A useful way to think of recursive functions is to imagine them as a process being performed where one of the instructions is to "repeat the process". This makes it sound very similar to a loop because it repeats the same code, and in some ways it is similar to looping.

On the other hand, recursion makes it easier to express ideas in which the result of the recursive call is necessary to complete the task. Of course, it must be possible for the "process" to sometimes be completed without the recursive call.

One simple example is the idea of building a wall that is ten feet high; if I want to build a ten foot high wall, then I will first build a 9 foot high wall, and then add an extra foot of bricks. Conceptually, this is like saying the "build wall" function takes a height and if that height is greater than one, first calls itself to build a lower wall, and then adds one a foot of bricks.

C PROGRAM TO FIND THE LARGEST ELEMENT IN AN ARRAY USING RECURSION.

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

```
#define MAX 100
```

```
int getMaxElement(int []);
```

```
int size;
```

```
int main(){
```

```
    int arr[MAX],max,i;
```

```
    printf("Enter the size of the array: ");
```



```

scanf("%d",&size);

printf("Enter %d elements of an array: ", size);
for(i=0;i<size;i++)
    scanf("%d",&arr[i]);

max=getMaxElement(arr);

printf("Largest element of an array is: %d",max);

return 0;
}

int getMaxElement(int arr[]){
    static int i=0,max=-9999;

    if(i < size){
        if(max<arr[i])
            max=arr[i];
        i++;
        getMaxElement(arr);
    }

    return max;
}

```

Sample output:

```

Enter the size of the array: 5
Enter 5 elements of an array: 1 4 5 6 2
Largest element of an array is: 6

```

4)

C PROGRAM TO SEPARATE EVEN AND ODD NUMBERS OF AN ARRAY AND PUT THEM IN TWO ARRAYS.

The program first finds the odd and even elements of the array. Then the odd elements of an array is stored in one array and even elements of an array is stored in another array.

```

/**
 * C program to put even and odd elements of an array in two separate array
 */

```

```

#include <stdio.h>

```

```

#define MAX_SIZE 1000 //Maximum size of the array

```

```

void printArray(int arr[], int len);

```

```

int main()
{
    int arr[MAX_SIZE], i, n;
    int even[MAX_SIZE], odd[MAX_SIZE], evenCount, oddCount;

    /*
     * Reads size and elements in the array
     */
    printf("Enter size of the array: ");
    scanf("%d", &n);
    printf("Enter elements in the array: ");
    for(i=0; i<n; i++)
    {
        scanf("%d", &arr[i]);
    }

    evenCount = oddCount = 0;
    for(i=0; i<n; i++)
    {
        // If arr[i] is odd
        if(arr[i] & 1)
        {
            odd[oddCount] = arr[i];
            oddCount++;
        }
        else
        {
            even[evenCount] = arr[i];
            evenCount++;
        }
    }

    printf("\nElements of even array: \n");
    printArray(even, evenCount);

    printf("\nElements of odd array: \n");
    printArray(odd, oddCount);

    return 0;
}

```

```

/**
 * Prints the entire integer array
 * @arr Integer array to be displayed or printed on screen
 * @len Length of the array

```

```

*/
void printArray(int arr[], int len)
{
    int i;
    printf("Elements in the array: ");
    for(i=0; i<len; i++)
    {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

```

Enter size of the array: 10
Enter elements in the array: 0 1 2 3 4 5 6 7 8 9

Elements of even array:
Elements in the array: 0 2 4 6 8

Elements of odd array:
Elements in the array: 1 3 5 7 9

5)

SPARSE MATRIX:

If the number of zeros in a matrix exceeds $(n*m)/2$, where n, m is the dimension of the matrix, matrix is sparse matrix. Sparse matrix has more zero elements than nonzero elements.

C PROGRAM TO DETERMINE A GIVEN MATRIX IS A SPARSE MATRIX.

```

/*
 * C Program to check if a Matrix is a Sparse Matrix
 */
#include <stdio.h>

```

```

void main ()
{
    int matrix[10][10];
    int i, j, m, n;
    int sparse_counter = 0;

    printf("Enter the order of the matix \n");
    scanf("%d %d", &m, &n);
    printf("Enter the elements of the matix \n");
    for (i = 0; i < m; ++i)
    {
        for (j = 0; j < n; ++j)
        {
            scanf("%d", &matrix[i][j]);
            if (matrix[i][j] == 0)

```

```

        {
            ++sparse_counter;
        }
    }
}
if (sparse_counter > ((m * n) / 2))
{
    printf("The given matrix is Sparse Matrix !!! \n");
}
else
    printf("The given matrix is not a Sparse Matrix \n");
printf("There are %d number of Zeros.", sparse_counter);
}

```

Output:

Enter the order of the matrix 3 3

Enter the elements of the matrix

1 2 3

4 0 0

0 0 0

The given matrix is Sparse Matrix !!!

There are 5 number of Zeros.

6)

Here we are setting up the pointer to the base address of array and then we are incrementing pointer and using * operator to get & sum-up the values of all the array elements.

C PROGRAM TO CALCULATE THE SUM OF ARRAY ELEMENTS USING POINTER.

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

```
int main()
```

```
{
```

```
    int array[5];
```

```
    int i,sum=0;
```

```
    int *ptr;
```

```
    printf("\nEnter array elements (5 integer values):");
```

```
    for(i=0;i<5;i++)
```

```
        scanf("%d",&array[i]);
```

```
    /* array is equal to base address
```

```
    * array = &array[0] */
```

```
    ptr = array;
```

```
    for(i=0;i<5;i++)
```

```
    {
```

```
        /* ptr refers to the value at address
```

```
        sum = sum + *ptr;
```

```
        ptr++;
```

```

    }

    printf("\nThe sum is: %d",sum);
}

```

Output:

Enter array elements (5 integer values): 1 2 3 4 5

The sum is: 15

7)

C PROGRAM TO APPEND THE CONTENTS OF A FILE AT THE END OF ANOTHER FILE WITHOUT USING ANY BUILT-IN FUNCTIONS.

```

/*
 * C Program to Append the Content of File at the end of Another
 */
#include <stdio.h>
#include <stdlib.h>

```

```

main()
{
    FILE *fsring1, *fsring2, *ftemp;
    char ch, file1[20], file2[20], file3[20];

    printf("Enter name of first file ");
    gets(file1);
    printf("Enter name of second file ");
    gets(file2);
    printf("Enter name to store merged file ");
    gets(file3);
    fsring1 = fopen(file1, "r");
    fsring2 = fopen(file2, "r");
    if (fsring1 == NULL || fsring2 == NULL)
    {
        perror("Error has occurred");
        printf("Press any key to exit...\n");
        exit(EXIT_FAILURE);
    }
    ftemp = fopen(file3, "w");
    if (ftemp == NULL)
    {
        perror("Error has occurs");
        printf("Press any key to exit...\n");
        exit(EXIT_FAILURE);
    }
    while ((ch = fgetc(fsring1)) != EOF)
        fputc(ch, ftemp);
    while ((ch = fgetc(fsring2)) != EOF)

```

```

        fputc(ch, ftemp);
    printf("Two files merged  %s successfully.\n", file3);
    fclose(fsring1);
    fclose(fsring2);
    fclose(ftemp);
    return 0;
}

```

Output:

Enter name of first file a.txt

Enter name of second file b.txt

Enter name to store merged file merge.txt

Two files merged merge.txt successfully.

8)

C PROGRAM TO CREATE A FILE CONTAINING STUDENT'S RECORDS AND ALSO GIVE A PROVISION TO UPDATE /MODIFY THE RECORDS TOO

```

/*
 * C Program to Create STUDENT RECORD AND UPDATE IT
 */

```

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define size 200

```

```

struct student
{
    int stuid;
    char *name;
}*stu1, *stu2;

```

```

void display();
void create();
void update();

```

```

FILE *fp, *fp1;
int count = 0;

```

```

void main(int argc, char **argv)
{
    int i, n, ch;

```

```

    printf("1] Create a Record\n");
    printf("2] Display Records\n");
    printf("3] Update Records\n");
    printf("4] Exit");
    while (1)
    {

```



```

printf("\nEnter your choice : ");
scanf("%d", &ch);
switch (ch)
{
case 1:
    fp = fopen(argv[1], "a");
    create();
    break;
case 2:
    fp1 = fopen(argv[1], "rb");
    display();
    break;
case 3:
    fp1 = fopen(argv[1], "r+");
    update();
    break;
case 4:
    exit(0);
}
}
}

/* To create an student record */
void create()
{
    int i;
    char *p;

    stu1= (struct emp *)malloc(sizeof(struct emp));
    stu1->name = (char *)malloc((size)*(sizeof(char)));
    printf("Enter name of student : ");
    scanf(" %[^\n]s", stu1->name);
    printf("Enter student id : ");
    scanf(" %d", &stu1->id);
    fwrite(&stu1->id, sizeof(stu1->id), 1, fp);
    fwrite(stu1->name, size, 1, fp);
    count++; // count to number of entries of records
    fclose(fp);
}

/* Display the records in the file */
void display()
{
    stu2=(struct student*)malloc(1*sizeof(struct stu));
    stu2->name=(char *)malloc(size*sizeof(char));
    int i = 1;

    if (fp1 == NULL)

```



```

    printf("\nFile not opened for reading");
while (i <= count)
{
    fread(&stu2->id, sizeof(stu2->id), 1, fp1);
    fread(stu2->name, size, 1, fp1);
    printf("\n%d %s", stu2->id, stu2->name);
    i++;
}
fclose(fp1);
free(stu2->name);
free(stu2);
}

void update()
{
    int id, flag = 0, i = 1;
    char s[size];

    if (fp1 == NULL)
    {
        printf("File cant be opened");
        return;
    }
    printf("Enter student id to update : ");
    scanf("%d", &stuid);
    stu2 = (struct student*)malloc(1*sizeof(struct student));
    stu2->name = (char *)malloc(size*sizeof(char));
    while(i<=count)
    {
        fread(&stu2->id, sizeof(stu2->id), 1, fp1);
        fread(stu2->name, size, 1, fp1);
        if (id == stu2->id)
        {
            printf("Enter new name of student to update : ");
            scanf("%[^\n]s", s);
            fseek(fp1, -204L, SEEK_CUR);
            fwrite(&stu2->id, sizeof(stu2->id), 1, fp1);
            fwrite(s, size, 1, fp1);
            flag = 1;
            break;
        }
        i++;
    }
    if (flag != 1)
    {
        printf("No student record found");
        flag = 0;
    }
}

```

```
fclose(fp1);
free(stu2->name);    /* to free allocated memory */
free(stu2);
}.
```

Output:

- 1] Create a Record
- 2] Display Records
- 3] Update Records
- 4] Exit

Enter your choice : 1

Enter name of student: aaa

Enter stu id : 100

Enter your choice : 1

Enter name of student: bbb

Enter stu id : 200

Enter your choice : 1

Enter name of student: ccc

Enter stu id : 300

Enter your choice : 1

Enter name of student: ddd

Enter stu id : 400

Enter your choice : 1

Enter name of student: eee

Enter stu id : 500

Enter your choice : 2

100 aaa

200 bbb

300 ccc

400 ddd

500 eee

Enter your choice : 3

Enter student id to student: 300

Enter new name of student to update : cdefgh

Enter your choice : 2

100 aaa

200 bbb

300 cdefgh

400 ddd

500 eee

Enter your choice : 4