Program:10

Sandy Samson

2047253

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

struct node {
char data;
struct node *next;
                                };
typedef struct node NODE;
NODE *head;

void convert(char* str);
void add(char);
void rev();
void display();
int main() {
        int ch;
        head = NULL;
        char string[50];
        printf("Sandy Samson\n");
        printf("2047253\n\n");
        printf("1.Enter a string\n");
        printf("2.Dispay string\n");
        printf("3.Reverse string\n");
        printf("4.exit\n\n");
```

```
while (1)
        {
        printf("Enter your choice : \n");
        scanf("%d", &ch);
        switch (ch)
        {
        case 1: printf("Enter the string : \n");
                        getchar();
                        fgets(string, 50, stdin);
                        convert(string);
                        printf("\n");
                        printf("The String:\n");
                        display();
                        break;
        case 2: printf("String:\n");
                        display();
                        break;
        case 3: printf("\noriginal String:\n");
                        display();
                        rev();
                        printf("Reversed String:\n");
                        display();
                        break;
        case 4:
                        exit(0);
                        break;
                        default:printf("\n invalid input");
                }
        }
```

```
}
void convert(char* str) {
for (int i = 0; i < strlen(str)-1; i++) add(str[i]);
        }
        void add(char item) {
        NODE *NewNode, *CurrentPtr;
        NewNode = (NODE *) malloc(sizeof(NODE));
        NewNode->data = item;
        NewNode->next = NULL;
        if (head == NULL) head = NewNode;
        else {
        CurrentPtr = head;
        while (CurrentPtr->next != NULL) CurrentPtr = CurrentPtr->next;
        CurrentPtr->next = NewNode;
        }
}
void rev() {
NODE *CurrentPtr, *PrevPtr, *NextPtr;
CurrentPtr = head;
PrevPtr = NULL;
NextPtr = NULL;
while (CurrentPtr != NULL) {
NextPtr = CurrentPtr->next;
CurrentPtr->next = PrevPtr;
PrevPtr = CurrentPtr;
CurrentPtr = NextPtr;
        }
head = PrevPtr;
}
```

```
        void display() {
        NODE *CurrentPtr;
        CurrentPtr = head;
        printf("%c", CurrentPtr->data);
        while (CurrentPtr->next != NULL) {
        CurrentPtr = CurrentPtr->next;
        printf(" %c", CurrentPtr->data);
        }
        printf("\n");
}
```