PSUC

Week - 7

V Satya Siva Lalitha Gayathri Boda
19131A05P1
CSE 4

Program 1

Aim: C program to convert upper case character to lower case and vice versa in a given string.

Program:

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
void main()
{
    char a[20], b[20];
    int n, i, j=0, c;
    gets(a);
    n = strlen(a);
    for(i=0; i<n; i++)
    {
        if(islower(a[i]))
        b[i] = toupper(a[i]);
        else if(isupper(a[i]))
        b[i] = tolower(a[i]);
    }
    b[i] = '\0'
    puts(b);
}
```

Program 2

Aim: C program to delete all vowels in a given string and display the remaining string.

Program:

```
#include <stdio.h>
#include <string.h>
void main ()
{
    char a[20], b[20];
    int n,i, j=0, c;
    gets(a);
    n = strlen(a);
    for (i=0; i<n; i++)
    {
      if (a[i]== 'A' || a[i]== 'E' || a[i]== 'I' || a[i]=='O'
          || a[i]== 'U' || a[i]=='a' || a[i]== 'e' ||
       a[i]== 'i' || a[i] == 'o' || a[i]== 'u')
      continue;
      else
      b[j]= a[i]
      j++;
    }
    puts(b);
}
```

Program 3

Aim: C program to check whether a given string is palindrome or not.

program.

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <string.h>
int main ()
{
    char s[50];
    char temp;
    char p[50];
    int i,j,c,r;
    gets (s);
    strcpy (p,s);
    for (i=0,j=c-1; i<c/2; i++,j--)
    {
        temp = s[i];
        s[i] = s[j];
        s[j] = temp;
    }
    if (strcmp (s,p) == 0)
    printf ("palindrome");
    else
    printf ("not palindrome");
}
```

Program 4.

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#include <stdbool.h>
int main()
{
    char a[10], b[10];
    int c,i;
    bool flag1 = true;
    bool flag2 = true;
    gets(a);
    fflush(stdin);
    gets(b);
    for (i=0, i< strlen(a); i++)
    {
      if (isdigit(a[i])==1 || a[i] == '\0')
      {
        flag1 = true;
        break;
      }
      else
        flag1 = false.
    }
    for (i=0; i< strlen(a); i++)
    {
      if(isdigit(b[i])==1 || b[i] = '\0')
      {
        flag2 = true;
        break;
      }
```

```
    else
     flag2= false;
}
if (flag1 == false || flag2 == false)
 printf ("Enter correct input");
 else
 printf ("%d", atoi (a) + atoi (b));
}
```

Week-8

Program

Aim: C program that implements the following user defined string handling functions.

- To find length of given string.
- To copy the contents of one string to another.
- To reverse the contents of a string.
- To compare 2 strings.
- To concatenate 2 strings.

Program:

```
#include <stdio.h>
#include <string.h>
int len(char[50]);
void copy(char[50]);
void rev(char[50]);
void comp(char[50], char[50]);
void concat(char[50], char[50]);
void main()
{
    int choice;
    char s[50], c[50];
    printf("1. length \n 2. copy string \n 3. reversing
    a string \n 4. comparing a string \n 5. concatenating
    strings");
    scanf("\n %d \n", &choice);
    fflush(stdin);
    switch(choice)
    {
        case 1:
```

```
        fflush(stdin);
        gets(s);
        int k = len(s);
        printf("string length is %d", k);
        break;
    case 2:
        gets(s);
        copy(s);
        break;
    case 3:
        gets(s);
        rev(s);
        break;
    case 4:
        gets(s)
        fflush(stdin);
        gets(c);
        comp(s,c);
        break;
    case 5:
        gets(s);
        fflush(stdin);
        gets(c);
        concat(s,c);
        break;
    default:
        printf("Enter right choice");
    }
}
int len(char n[50])
{
    int i=0;
    while(n[i] != '\0')
    i++;
    return i;
}
```

19131A05P1

```
copy (char n[50]
{
   int i=0;
   char cp[50];
   int length = len(n);
   while (n[i] != '\0')
   {
      cp[i] = n[i];
      i++;
   }
   cp[i] = '\0';
   printf ("string copied is ");
   puts (cp);
}
rev (char n[50])
{
   int i=0, j, t;
   char r[50]
   j = len(n) - 1;
   for (i=0; i < len(n); i++)
   {
      r[i] = n[j]
      j--
   }
   r[i] = '\0';
   printf (" string reversed is ");
   puts(r);
}
comp (char m[50], char n[50])
{
   int i=0, c=0;
   if (len(m) == len(n))
   {
```

```
    while (m[i] != '\0')
    {
       if (n[i] == m[i])
        c++;
        i++;
     }
  }
  if (c==0)
  printf ("strings are not same");
  else
  printf ("strings are same");
}
concat (char n[50], char m[50])
{
   int i=0;
   int j=0;
   int l = len(n) + len(m);
   for (i=0; n[i] != '\0'; i++)
    n[i] = n[i];
    for (j=0; m[j] != '\0'; j++)
      n[i] = m[j];
       i++;
       }
      n[i] = '\0';
      puts (n);
      printf (" is concatenated string");
   }
```

Program: C program to demonstrate the usage of 10 predefined functions (String handling functions).

```
#include <stdio.h>
#include <string.h>
void main()
{
    int ch, n;
    char s[50], c[50], z;
    printf("1. length \n 2. copy \n 3. concat \n 4. nconcat
    \n 5. atoi \n 6. strchr \n 7. strncpy \n 8. strncmp
    \n 9. strcmp \n 10. atof \n");
    printf("Scan a number untill 0 is entered \n");
label : scanf("%d\n", &ch);
    switch(ch)
    {
        case 0:
            fflush(stdin);
            printf("process stopped");
            exit(1);
        case 1:
            gets(s);
            printf("Length of string is %d",
                                    strlen(s));
            break;
        case 2: gets(s);
            puts(strcpy(c,s));
```

```
        break;
case 3 :
        gets (s);
        fflush (stdin);
        gets(c);
        puts (strcat (s,c));
        break;
case 4 :
        scanf ("\n%d\n", &n);
        gets (s);
        fflush (stdin);
        gets (c);
        puts (strncat (s,c,n));
        break;
case 5 :
        gets(s);
        fflush (stdin);
        gets (c);
        printf ("Sum of both strings %d\n",
                                atoi(s) + atoi (c));
        break;
case 6 : scanf ("%c\n", &z);
        gets(s)
        puts (strchr (s,z));
        break;
case 7 : gets (s);
        fflush (stdin);
```

```
        strncpy(c,s,5);
        printf("copied string : %.5s\n", c);
        break;
case 8: fflush(stdin);
        gets(s);
        fflush(stdin);
        gets(c);
        fflush(stdin);
        int res = strncmp(s,c,3);
        if (res < 0)
          printf("str1 is less than str2");
        else if (res > 0)
          printf("str2 is less than str1");
        else
          printf("str1 and str2 are equal");
        break;
case 9 : gets(s);
        fflush(stdin);
        gets(c);
        if (strcmp(s,c)==0)
          printf("Strings are same");
        else
          printf("strings are different");
        break;
```

```
case 10:
        gets(s);
        printf("floating value of string is
                                %f", atof(s));
        break;
default : printf("Enter correct choice");
        break;
}
while(ch! = 0)
{
  printf("\n Enter another number of your
                                choice");
  goto label;
}
}
```

Week-9

Program 1 : C program to demonstrate the usage of pointers.

```
#include <stdio.h>
int main()
{
    int n, *ptr = &n;
    scanf("%d", &n);
    printf("Address of a is %u \n", &n);
    printf("Address of a is %u \n", ptr);
    printf("Value of a is %d \n", n);
    printf("Value of a is %d \n", *ptr);
    return 0;
}
```

Program 2 : C program that uses dynamic memory allocation function to add n elements & display their average.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int n, S = 0;
    int *ptr;
    scanf("%d", &n);
    ptr = (int *) malloc (n * sizeof(int));
    if (ptr == NULL)
    {
```

```
    printf ("unable to allocate memory");
    exit (0);
  }
  printf (" Numbers are \n");
  for (int i = 0; i < n ; i++ ; ptr++)
  {
      scanf ("%d", ptr+i);
      s = s+ * (ptr+i);
  }
  printf (" sum is %d", s);
  free (ptr);
}
```

Program 3: C program that performs pointer arithmetic

```
#include <stdio.h>
#include <stdlib.h>
void main ()
{
    int arr[4] = {1,2,3,4};
    int b = 10;
    int * ptr1 = arr , * ptr2 = &b;
    printf ("%u \n" , ptr1);
    printf ("Variable address after pointer
     addition is %u \n" , (ptr1 + 2));
    printf ("Variable value after pointer addition
     %d \n" , * (ptr1 + 2));
    ptr1 + = 3;
    printf ("Address of variable after pointer
    subtraction %u \n" , (ptr1 - 2));
```

```
printf ("value of variable after pointer
        subtraction is %d \n" ,*(ptr-2));
printf (" Address of variable after pointer increment
        is %u \n", ptr++);
printf (" value of variable after pointer increment
        is %d \n" ,(*ptr++));
printf (" Address of variable after pointer
        decrement is %u \n", ptr--);
printf (" value of variable after pointer
        decrement is %d \n", *(ptr--));
if ( ptr1 > ptr2)
 printf (" ptr1 is greater than ptr2 \n");
 else if ( ptr2 > ptr1)
  printf (" ptr 2 is greater than ptr1 \n");
  else
   printf (" Both pointers are equal");
}
```

Program 4 : C program that implements call by reference.

```
#include <stdio.h>
void fun (int *, int *, int *);
void main ()
{
 int a, b, sum;
 scanf ("%d %d ", &a, &b);
 fun (&a, &b, &sum);
 printf (" Sum of a and b is %d" , sum);
}
```

```
void fun (int * p1, int * p2, int * s)
{
    * s = *(p1) + *(p2);
}
```

Week 10 Program 1

```
#include <stdio.h>
void fun1 ();
void fun2 ();
void fun3 ();
void fun4 (int);
int main ()
{
    int ch, n;
    void (*ptrfun) (int) = &fun4;
    printf ("1. pointer to pointer \n 2. pointer to
    array \n 3. pointer to p array of pointers \n
    4. pointer to functions \n");
    scanf ("%d", &ch);
    switch (ch)
    {
     case 1:
        fun1 ();
        break;
     case 2:
        fun2 ();
        break;
```

```
        case 3 :
                fun3 ();
                break;
        case 4 :
                printf (" Enter a number \n");
                scanf (" %d", &n);
                (*ptrfun) (n);
                break;
        default :
                printf (" Enter correct choice ");
    }
}
void fun1 ()
{
    int a, *p = &a, **ptr = &p;
    printf (" Enter a number \n");
    scanf (" %d", &a);
    printf (" Value of 'a' using pointer of pointers
     is %d \n", **ptr);
     printf (" Address of a using pointer of
     pointers is %u \n", *ptr);
}
void fun2 ()
{
    int a [20], i, n;
    int *ptr = a;
     printf (" Enter a number \n");
```

```
scanf ("%d", &n);
printf ("Enter elements of array");
for (i=0, i<n; i++)
scanf ("%d", (ptr+i));
printf ("Elements of array are \n");
for (i=0; i<n; i++)
printf ("%d", *(ptr+i));
}
void fun3 ()
{
  int a[10], *ptr[10], n;
  printf ("Enter a number");
  scanf ("%d", &n);
  printf ("Enter elements of array\n");
  for (int i=0; i<n; i++)
  {
    scanf ("%d", &a[i]);
    ptr[i] = &a[i];
  }
  printf ("Elements of array are \n");
  for (int j=0; j<n; j++)
  {
    printf ("%d", (*(ptr[j])));
  }
}
void fun4 (int n)
{
```

```
printf ("value of n using pointer to function
  is %d", n);
}
```

Week 11

program : C program to access & display the members of the structure.

```
#include <stdio.h>
#include <string.h>
struct student
{
  char name[100];
  int regno, m2, psuc, phy, stld;
};
int main()
{
    struct student s;
    printf ("Enter name of student \n");
    fflush (stdin);
    gets(s.name);
    printf ("\n Enter Reg.no , m2 , psuc, phy, stld
    marks of student \n");
    scanf ("%d %d %d %d %d", &s.regno, &s.m2,
                    &s.psuc, &s.phy, &s.stld);
    printf ("\n Details of student are \n");
      printf ("\n name : ");
      puts (s.name);
```

```
printf ("regno : %d", s.regno);
printf ("m2 marks : %d \n psuc marks: %d \n
    phy marks : %d \n stld marks : %d \n", s.m2,
        s.psuc, s.phy , s.stld);
}
```

Program 2: C program that demonstrates different ways to access the structure elements using pointers.

```
#include <stdio.h>
#include <string.h>
struct student
{
  char name [100];
  int regno, m2, psuc, phy, stld;
};
int main ()
{
    struct student *p1, s;
    p1 = &s;
    printf ("Enter student's name");
    gets (p1 → name);
    printf ("Enter reg.no, m2, psuc, phy, stld marks");
    printf ("\n");
    scanf ("%d", &p1 → regno);
    scanf ("%d", &p1 → m2);
    scanf ("%d", &p1 → psuc);
    scanf ("%d", &p1 → phy);
    scanf ("%d", &p1 → stld);
```

```
printf("\n Data of student is \n");
printf ("Name of the student is ");
puts (p1 -> name);
printf ("Registration number is %d \n", p1 -> regno);
printf ("Maths 2 marks are %d \n",((* p1). m2));
printf ("PSUC marks are %d \n",((* p1). psuc));
printf ("Physics marks are %d \n",((* p1). phy));
printf ("SLLd marks are %d \n",((* p1). stld));
}
```

Week 12

1) Program 1 : C program to read contents of a file and display on the output screen.

```
#include <stdio.h>
void main()
{
FILE *fp;
char ch;
fp = fopen ("content.txt","r");
if (fp == NULL)
{
  printf ("File can't be opened");
  exit(1);
}
```

```
while (1)
{
 ch = fgetc (fp);
  if (ch == EOF)
  break;
  printf ("%c", ch);
  }
  fclose (fp);
  }
```

Program 2: C program to copy contents of one file to another.

```
#include <stdio.h>
 void main ()
 {
    FILE *fp1, *fp2;
    char ch;
    fp1 = fopen ("content1.txt", "r");
    fp2 = fopen ("Newfolder.txt", "w");
    if (fp1 == NULL || fp2 == NULL)
    {
       printf (" Cant open file ");
       exit (1);
    }
    while (1)
    {
      ch = fgetc(fp1);
       if (ch == EOF)-
         break;
```

```
fputs (ch, fp2);
}
fclose (fp1);
fclose (fp2);
}
```

Program 3: C program to count and display the number of characters, words, lines in a file.

```
#include <stdio.h>
#include <stdlib.h>
void main ()
{
FILE *fp;
char ch;
int cc=0, wc=0, lc=0;
fp= fopen ("file.text","r");
if (fp== NULL)
{
    printf ("File cant be opened");
    exit(1);
}
while (1)
{
ch= fgetc(fp);
if (ch== EOF)
  break;
if (ch==' ' || ch== '\t' || ch== '\n' || ch== '\0')
wc++;
if (ch== '\n' || ch== '\0')
lc++;
```

```
        cc++;
    }
    if (characters > 0)
    {
      wc++;
      lc++;
    }
    printf("There are %d characters in file\n", cc);
    printf("There are %d words in file\n", wc);
    printf("There are %d lines in file\n", lc);
}
```

4. C program to print last n characters of a file by reading file name, n values from command line.

```
#include <stdio.h>
int main(int argc, char *argv[])
{
  FILE *fp; int n;
  char ~~str[100];~~ c;
  if (argc < 2)
  {
    printf("No arguments supplied");
  }
  fp = fopen(argv[2], "r");
  n = atoi(argv[1]);
  fseek(f1, -n, SEEK_END);
  while ((c = fgetc(fp)) != EOF)
  {
    printf("%c", c);
  }
  fclose(fp);
}
```