Experiment Name : SHELL INPUT/OUTPUT REDIRECTION AND GCC COMPILE COMMANDS

Problem Statement :

I> REDUCTION

① OUTPUT REDIRECTION

$ ls > dalbfile.txt

// appends listing of ls command into

$ ls -l > dalbfile.txt

// long listing is appended.

$ cat > pes.txt // prints the contents

$ cat pes.txt > dalbfile.txt

//adds contents of first file to second

② INPUT REDUCTION

$ wc < dalbfile.txt

// displays the no of lines, words & char in

③ ERROR REDIRECTION

$ gcc a.c 2>error.txt

/* redirects error into error.txt */

Std i/p → 0
Std o/p → 1
Std error → 2

③ DISCARD THE OUTPUT

$ command >/dev/null

Problem Statement :

Week

9.8

9035123456 78

25000

Name: Pramod M.N

SSN: 123456

Age : 28

Gender: M

Performance Rating: 9.800000

Adhaar Number: 903512345678

Salary: 25000

(3) WAP ~~to~~ using bitwise operators for the follow

(a) Check a bit

(b) Set a specified bit and print the result

(c) Clear a specified bit and print the result

(a)

```
#include<stdio.h>
int main()
{
  int n, pos;
  printf("Enter the number and position");
  scanf("%d %d", &n, &pos);
  n = n & (1<<pos);
  printf("n: %d", n);
```

55

OUTPUT:

Enter the number and position: 9 01

n: 0

(b)
```
#include <stdio.h>
int main()
{
 int n, pos;
 printf("Enter n and pos: ");
 scanf("%d %d", &n, &pos);
 printf("n: ", n | (1 << pos));
 return 0;
}
```

OUTPUT:

Enter n and pos: 8 0

n: 9

(c)
```
#include<stdio.h>
int main()
{
 int n, pos;
 printf("Enter n and pos: ");
 scanf("%d %d", &n, &pos);
 printf("n: ", n ^ (1 << pos));
 return 0;
}
```

OUTPUT:

Enter n and pos: 15 2

n: 11

Problem Statement :

WAP to calculate the time taken for execution of a C Program.

```
#include <stdio.h>
#include <time.h>
int main()
{
  int n1, n2;
  clock_t start;
  start = clock();
  printf("Enter two numbers: ");
  scanf("%d %d", &n1, &n2);
  printf("The sum: %d\n", n1 + n2);
  printf("The time taken : %lf \n",((double)clock() - start)/CL
  return 0;
}
```

OUTPUT

```
Enter two numbers: 10 20
The sum: 30
The time taken : 0.00170
```

Problem Statement :

Week No.

③ WAP to generate the pattern shown:

```
1 = 1
1 + 2 = 3
1 + 2 + 3 = 6
```

```
#include <stdio.h>
int main()
{
  int n, sum, i, j;
  printf("Enter n: ");
  scanf("%d", &n);
  for (i = 1; i <= n; i++)
  {
    sum = 0;
    for (j = 1; j < i; j++)
    {
      sum += j;
      printf("%d + ", j);
    }
    printf("%d = %d\n", j, sum + j);
  }
  return 0;
}
```

OUTPUT:

```
Enter n: 4
1 = 1
1 + 2 = 3
1 + 2 + 3 = 6
1 + 2 + 3 + 4 = 10
```

mab
a

server.c

```
#include "head.h"
void reverseArrayP (int *x, int n)
{
   int i;
   for (i = 1; i <= n; i++)
       printf ("%d", *(x+n-i));
   printf ("%c", '\n');
}
void reverseArray1 (int x[], int n)
{
  int i;
  for (i = 1; i <= n; i++)
      printf ("%d", x[n-i]);
  printf ("%c", '\n');
}
```

head.h

```
#include <stdio.h>
void reverseArray1 (int x[], int n);
void reverseArrayP (int *x, int n);
```

client.c

```
#include "head.h"
int main ()
{
  int a[20], n, i;
  printf ("Enter n: ");
  scanf ("%d", &n);
  printf ("Enter the elements:");
  for (i = 0; i < n; i++)
      scanf ("%d", &a[i]);
```

Experiment Name :	Expt. No.
	Date
Problem Statement :	
	Week No.

```
    printf("The elements in reverse order (using arrays): ");
    reverseArray1(a, n);
    printf("The elements in reverse order (using pointers): ");
    reverseArrayP(a, n);
    return 0;
}
```

makefile

```
a.out : client.o server.o
        gcc client.o server.o
client.o : client.c head.h
        gcc -c client.c
server.o : server.c head.h
        gcc -c server.c
```

OUTPUT:

```
Enter n: 5
Enter the elements: 1 2 3 4 5
The elements in reverse order (using arrays): 5 4 3 2 1
The elements in reverse order (using pointers): 5 4 3 2 1
```

Problem Statement :

26/02/18

Week No.

① Write a functions to ~~reverse~~

ⓐ reverse a string

ⓑ check for equality of two strings

Use these functions in client to check if a string is palindrome

client.c

```
#include "header.h"
int main()
{
  char str[25];
  printf("Enter a string: ");
  scanf("%s", str);
  int res = strrev(str);
  if (res == 1)
     printf("The string is a palindrome");
  else
     printf("The string isn't a palindrome");
  return 0;
}
```

header.h

```
#include<string.h>
#include<stdio.h>
int strrev(char a[]);
int strcom(char a[], char b[]);
```

server.c

```
#include "header.h"
int strrev (char a[])
{
   int len = strlen(a), i;
   char b[25];
   for (i = len-1; i >= 0; i--)
         b[len-i-1] = a[i]
   return strcom(a, b);
}
int strcom (char a[], char b[])
{
   int len = strlen(a), i;
   for (i = 0; i < len; i++)
   {
      if (a[i] != b[i])
         return 0;
   }
   return 1;
}
```

makefile

```
a.out : client.o server.o
    gcc client.o server.o
client.o : client.c header.h
    gcc -c client.c
server.o : server.c header.h
    gcc -c server.c
```

OUTPUT:

```
Enter a string : gadag
The string is a palindrome
```

Experiment Name :	Expt. No.
	Date
Problem Statement :	
	Week No.

WAP to concatenate n strings

client.c

```
#include "header.h"
int main()
{
  int nos, i;
  char a[10][10], mains[100];
  printf("Enter the number of strings: ");
  scanf("%d", &nos);
  for(i=0; i<nos; i++)
     scanf("%s", a[i]);
  int len = strlen(a[0]);
  for(i=0; i<len; i++)
       mains[i] = a[0][i];
  for(i=1; i<nos; i++)
       strcat(mains, a[i]);
  printf("%s", mains);
  return 0;
}
```

header.h

```
#include <stdio.h>
#include <string.h>
void strcat(char *a, char *b);
```

server.c

```
#include "header.h"
void strconcat (char *a, char *b)
{
  int len = strlen(a), i = 0;
  while (*(b+i))
  {
    *(a+len+i) = *(b+i);
    i++;
  }
}
```

makefile

```
a.out: server.o client.o
    gcc server.o client.o
server.o: server.c header.h
    gcc -c server.c
client.o: client.c header.h
    gcc -c client.c
```

OUTPUT:

```
Enter the number of strings: 2
hi
bye
hibye
```

③ WAP to find all occurances of every character in a word

client.c

```
#include "header.h"
```

Experiment Name :	Expt. No.
	Date
Problem Statement :	
	Week No.

```
int main()
{
  char word[25], dl[10];
  printf("Enter a string: ");
  scanf("%s", word);
  int i = 0;
  while(word[i])
  {
    noOfOccurances(word, word[i], dl);
    i++;
  } return 0;
}
```

header.h

```
#include <stdio.h>
#include <string.h>
void noOfOccurances(char a[], char b, char *dl);
int charInDL(char b, char *dl);
```

server.c

```
#include "header.h"
void noOfOccurances(char a[], char b, char *dl)
{
  if(charInDL(b, dl))
  {
    int i = 0; count = 0;
    while(a[i])
    { if(a[i] == b)
        count++;
```

```
        i++;
      }
      printf("%c %d\n", b, count);
    }
}
int charInDL(char b, char *dl)
{
  int i=0;
  while (*(dl+i))
  {
    if (b==*(dl+i))
      return 0;
    i++;
  }
  return 1;
}
```

makefile

```
a.out : server.o client.o
        gcc server.o client.o
server.o : server.c header.h
        gcc -c server.c
client.o : client.c header.h
        gcc -c client.o
```

OUTPUT:

```
Enter a word: word
w: 1
```

Problem Statement :

WAP to find leftmost and rightmost occurance of a charaders. Use these functions to find position of every occurance in a string

client.c

```
#include "head.h"
int main()
{ int m; i;
  scanf ("%d", &m);
  char s[m];
  scanf ("%s", s);
  for (i = 0; i < m; i++)
  { int j, k;
    j = lm (s, s[i], m);
    k = rm (s, s[i], m);
    printf ("The leftmost occurance is %c %d", s[i], j);
    printf ("The rightmost occurance is %c %d", s[i], k);
  }
}
```

head.h

```
#include <stdio.h>
int lm (char s[], char t, int m);
int rm (char s[], char t, int m);
```

.c

```
#include "head.h"
int lm(char s[], char t, int n)
{
    int a, i;
    for (i = 0; i < n; i++)
    {
        if (t == s[i])
        {
            a = i + 1;
            break;
        }
    }
    return a;
}
int rm(char s[], char t, int n)
{
    int i;
    for (i = n-1; i >=0; i--)
    {
        if (t == s[i])
        {
            return i+1;
        }
    }
    return 0;
}
```

makefile:

```
a.out : server.o client.o
        gcc server.o client.o
server.o : server.c head.h
        gcc -c server.c
client.o : client.c head.h
        gcc -c client.c
```

OUTPUT

```
a
aravinath
```

Experiment Name : STRUCTURE, ARRAY OF STRUCTURES AND POINTERS	
	Date
Problem Statement :	12/03/18
	Week No.
	8

① WAP to compare two dates and print appropriate message using structures

server.c

```
#include "head.h"
int dateCompare (struct date *obj1, struct date *obj2)
{
 if (obj1->d == obj2->d && obj1->m == obj2->m && obj1->y == obj2->y)
        return 1;
  else
        return 0;
}
```

head.h

```
#include <stdio.h>
struct date1
{
   int d, m, y;
} d1;
struct date2
{
   int d, m, y;
} d2;
int dateCompare (struct date1 *obj1, struct date2 *obj2);
```

client.c

#in

client.c

```
#include "head.h"
int main()
{ int res;
  printf("Enter the first date (dd mm yyyy) \n");
  scanf("%d%d%d", &d1.d, &d1.m, &d1.y);
  printf("Enter the second date (dd mm yyyy) \n");
  scanf("%d%d%d", &d2.d, &d2.m, &d2.y);
  res = dateCompare(&d1, &d2);
  if (res == 1)
     printf("The dates are equal \n");
  else
     printf("The dates are unequal \n");
  return 0;
}
```

makefile

```
a.out : server.o client.o
    gcc server.o client.o
server.o : server.c head.h
    gcc -c server.c
client.o : client.c head.h
    gcc -c client.c
```

OUTPUT:

```
Enter the first date (dd mm yyyy)
8 12 1999
Enter the second date (dd mm yyyy)
8 12 1999
The dates are equal
```