INDEX

Sl. No.	Date	Name of the Experiment	Page No.	Remark
		PART-A		
01.	2/12/22	Linear Search	1	
02.	20/1/23	Bubble Sort	2	
03.	16/12/22	Palindrome	3	
04	9/12/22	Compute ncr using recursive procedure	4-5	
05.	6/1/23.	Read current time and date from system & display	6-7	
06.	18/11/22	Display alphabets on screen	8	
07.	18/11/22	Factorial	8	
08.	2/12/22	Fibonacci series	9	
09.	25/11/22	Convert binary to BCD equivalent	10	
10.	25/11/22	Count the numbers of 0's & 1's	10	
		PART-B		
01.	27/1/23	Half sine wave.	11-1	
02.	10/2/23	Implementing the buzzer.	11-12	
03.	17/2/23	Drive stepper motor interface and rotate the motor.	13-14	
04.	3/2/23	LED display on screen.	14	
05.		Interface 4*4 matrix keyboard		
05.	3/2/23.	Binary display code.	15.	

Software Programs

Part-A

Exp No. 1 :-

Linear Search

Algorithm:

Step 1 : Select the first element as the current element.

Step 2 : Compare the current element with the target element. If matches then go to step 5.

Step 3 :- If there is a next element, then set current element to next element & go to step 2.

Step 4 : Target element not found. Goto to step 6.

Step 5 :- Traget element found & return location

Step 6 : Exit process.

```
.model small
print macro msg
lea dx, msg
mov ah, 09h
int 21h
endm

data segment
array dw 1111h, 2222h, 3333h, 3344h, 4455h
len equ ($-array)
key dw 2221h
msg1 db "key found $"
msg2 db "key not found $"
data ends
```

```
Code Segment
assume cs: code, ds: data
start : mov ax, data
        mov ds, ax
        lea si, array
        mov cx, len
        mov bx, key
next : mov ax, [si]
       cmp ax, bx
       je found
       inc si
       dec cx
       jnz next
       jne nf
   nf: print msg1
       jmp exit
found : print msg1
exit : mov ah, 4ch
       int 21h
       code ends
       end start.
```

Output :

Input	Output
Enter the number of elements in the array: 5 Enter the array elements : Search element :	Element available at location 2

Exp No. 2 -

Bubble Sort

Algorithm :

Step 1 : Declare the array with the numbers that need do be sorted

Step 2 : Initialize iteration count (n-1)

Step 3 : Initialize comparison counter

Step 4 : Compare num1 and num2

Step 5 : Num1 <= num2 do not exchange

step 6 : Num1 >= num2 then exchange the number position

step 7 : Decrement iteration counter, comparison counter

step 8 : Terminate the program.

Program :

```
.model small
data segment
org 1000h
list dw 30h, 20h, 10h
count equ 3
data ends

code segment
Assume cs: code, ds: data
start: mov ax, data
       mov ds, ax
       mov dx, count - 1

again 0: mov cx,dx
         mov si, offset list
again 1: mov ax, [si]
         cmp ax, [si+2]
         jl por1
```

```
        xchg [si+2], ax
        xchg [si], ax
pr1 : add si, 02
      loop again1
      dec dx
      jnz again0
      mov ah, 4ch
      int 21h
code ends
end start.
```

Output:

At source before execution	11	33	99	22	44.

Exp No. 3 –

Palindrome

Algorithm :-

Step 1 : Create display macro to display the message.
Step 2 : Declare the string
Step 3 : Declare the message to display
Step 4 : Find the reverse of string and store in string 1
Step 5 : Is string = string 1, display it is palindrome
Step 6 : Else if display not a palindrome.
Step 7 : Terminate the program.

Program :-

```
.model small
data segment
s db "nitin"
l dw $-s
r db 10 dup(?)
m1 db "palindrome $"
m2 db "not palindrome $".
data ends.

code segment
assume cs:code, ds:data
start : mov ax, data
        mov ds, ax
        mov cs, ax
        mov cx, l
        lea si, s
```

```
        lea di, rs
        add di, cx
        dec di
b :     mov al, [si]
        mov [di], al
        inc si
        dec di
        loop b
        lea si, s
        lea di, rs
        mov cx, l
        repe cmpsb
        jne np
        lea dx, m1
        mov ah, 09h
        int 21h
        jmp d
np :    lea dx m2
        mov ah, 09h
        int 21h
d:      mov ah, 4ch
        int 21h
        code ends
        end start.
```

Output :-

Input	Output
Nitin	palindrome
Hello	Not palindrome

Exp No. 4 –

Compute ncr using recursive procedure.

Algorithm :-

Step 1 : Initialize the values for n, r, res

Step 2 : Call ncr procedure

Step 3 : If r=0, res=1 goto step

Step 4 : Else r=r-1

Step 5 : Substract n-r. Multiply (n-r)*res

Step 6 : Res = (n-r)* res/2

Step 7 : Return to step 2 save the result in res

Step 8 : Terminate the program.

Program :-

```
.model small
data segment
s1 db "enter n: $"
s2 db 10, 13, "enter r: $"
s3 db 10, 13, "ncr : $"
n db ?
r db ?
nn db ?
rr db ?
diff db ?
data ends.
code segment
assume cs: code, ds: data
start :
```

```
mov ax, data
mov ds, ax
lea dx, s1
mov ah, 09h
int 21h
mov ah, 1h
int 21h
sub al, 30h
mov n, al
mov ch, 0h
mov cl, n
mov ax, 1h
call fact
mov nn, al
lea dx, s2
mov ah, 09h
int 21h
mov ah, 01h
int 21h
sub al, 30h
mov r, al
mov ah, n
cmp ah, al
jb fe
mov ch, 0h
mov cl, r
mov ax, 01h
call fact
mov rr, al
mov ah, n
mov al, r
sub ah, al
```

```
mov diff, ah
mov ax, 1h
mov ch, 0h
mov cl, diff
call fact
mov cl, rr
mul cl
mov cl, al
mov al, nn
div cl
aam
mov bx, ax
lea dx, s4
mov ah, 09h
int 21h
add bx, 3030h
mov bdl, bh
mov ah, 02h
int 21h
mov dl, bl
mov ah, 02h
int 21h
jmp exit
fe:
lea dx, s3
mov ah, 09h
int 21h
exit:
mov ah, 4ch
int 21h
fact proc
```

```
cmp cl, 0h
je f0
f:
mul cl
dec f
out
f0:
out
fact ends
code ends
end start.
```

Output :-

For the value N=4, R=2

Result is = 6.

Exp No. 5 –

Read the current time and date from system and display.

Program :-

```
• model small
• stack
• data
m1 db "current time : $"
m2 db 10, 13, "current date : $"
hr db ?
min db?
s db ?
day db ?
month db ?
year dw ?
• code
mov ax, @data
mov ds, ax
mov ah, 2ch
int 21h
mov hr, ch
mov min, cl
mov s, dh
mov ah, 2ah
int 21h
mov day, dl
mov month, dh
mov year, cx
lea dx, m1
mov ah, 09
```

```
int 21h
mov cl, hr
mov ch, 0
call disp
mov dl, ':'
mov ah, 2
int 21h
mov cl, min
mov ch, 0
call disp
mov dl, ':'
mov ah, 02
int 21h
mov cl, s
mov ch, 0
call disp
lea dx, m2
mov ah, 09
int 21h
mov cl, day
mov ch, 0
call disp
mov dl, '/'
mov ah, 02
int 21h
mov cl, month
mov ch, 0
call disp
mov dl, '/'
```

```
mov ah, 02
int 21h
mov cx, year
call disp
mov ah, 4ch
int 21h
disp proc
mov bx, 0
n: mov al, bl
add al, 1
daa
mov bl, al
jnc n1
add al, 1
daa
mov bh, al
n1: loop n
mov dl, bl
and dl, 0f0h
mov cl, 4
shr dl, cl
add dl, 30h
mov ah, 02
int 21h
mov dl, bl
and dl, 0fh
```

```
add dl, 30h
int 21h
out
disp endp
end.
```

Output :-

Date : 23/07/2022

Time : 10:15:58

Expt No. 6 -

Display alphabets on screen.

Program :-

```
.model small
code segment
assume cs: code ds: data
start : mov cx, 26
        mov dx, 65
    b:
    mov ah, 2
    int 21h
    inc dx
    loop b
    mov ah, 4ch
    int 21h
    inc dx
    mov ah, 4ch
    int 21h
    code ends
    end start.
```

Output :-

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Expt No. 7 -

Factorial

```
• madul small
data segment
n dw 5
data ends
code segment
assume cs: code . ds: data
start : mov ax, data
        mov ds, ax
        mov ax, 0000h
        mov ax, n
        mov cx, n
        dec cx
    b : dec n
        mul n
        dec cx
        mov ~~dx~~, ah, 4ch
        int 21h
    code ends
    end start .
```

Output :-

AX = 4C78

BX = 0000

CX = 0000

DX = 0078.

Expt No. 8 –

Fibonacci Series

Program :-

```
• model small
data segment
org 4000h
num db 07
org 5000h
fibo db 7
data ends

code segment
assume cs: code , ds : data
main : mov ax, data
       mov di , fibo
       mov dl , num
       mov al , 00h
       mov [di] al ; mov 00
       add di, 0h
       daa

       mov bL, dh
       mov [di], bl , mov 0
  Op : add al , bl
       daa
       add di , 01h
       mov [di], akl

      X chg al, bl
      dec dl.
```

```
jnp up
mov ah, 4ch
int 21h
code ends
end start.
```

Output:-

```
d 4000 ... 07
d 5000 ... 00 01 02 03 05 08 13 - 21
```

Expt No. 09 -

Convert binary number to BCD equivalent.

Program :-

```
.model small
data segment
org 1000h
n1 db 19h
n2 db ?
data ends
code segment
assume cs:code, ds:data
start : mov ax, @data
        mov ds, ax
        mov al, 00h
        mov al, n1
        mov bl, al
        shr al, 01
        x or al, bl
        mov n2, al
        mov ah, 4ch
        int 21h
code ends
end start.
```

Output

AX = 4C15

Expt No. 10 -

Count the numbers of 0's & 1's

```
.model small
Data segment
 n dw 1000h
 zero dw 01 dup(?)
 one dw 01 dup(?)
 data ends
 code segment
  assume cs: code, ds: data
  start : mov ax, data
  mov ds, ax
  mov ax, n
  mov bx, zero
  mov ax, one
  mov cx, 10h
  l;
  rol ax, l
  Jc 0
  inc bx
  cmp nx
  0;
  inc dx
  next : dec cx
  Jnz l
  mov zero, bx
  mov one, dx
 mov ah, 4ch
 int 21h
 code ends
 end start
```

Output :-

BX = 000F
DX = 0001

Part-B

1. Write a C program to execute half sin wave using 8051 microcontroller

Program :-

```
#include <intrins.h>
#include <reg 51.h>
main()
{
unsigned int i;
unsigned int w[16] = {128, 224, 255, 240, 192, 128, 64, 32, 16,
                      0, 16, 16, 22, 32, 64};

while (1)
{
 for(i=0; i<16; i++)
 {
   P1 = w[i];
   ~~no~~ _nop_ ();
 }
}
}
```

2. Design & develop an ALP to implement the buzzer using ARM TTDMI / LPC 2148.

Program:-

```
#include <LPC21XX.h>
void delay (unsigned long cnt)
{
unsigned long temp, val;
for (temp = 0 ; temp < cnt ; temp++)
for (val = 0; val < 10 ; val++)
}
int main (void)
{
unsigned int i;
IO0DIR = 0x00000200 ;
IO1DIR = 0x02000000;
while (1)
{
for (i = 0 ; i < 10000 ; i = i+100)
{
IO0SET = 0x00000200;
IO1SET = 0x2000000 ;
delay (i*10);
IO0CLR = 0x000002000;
IO1CLR = 0x02000000;
delay (i*10);
}
```

```
for (i = 10000; i > 0; i = i - 100)
{
IO0SET = 0x00000200;
IO1SET = 0x02000000;
delay (i * 10);
IO0CLR = 0x00000200;
IO1CLR = 0x02000000;
delay (i * 10);
}
}
}
```

3. Design and develop an assembly language program to drive stepper motor interface and rotate the motor by clock wise and anti clock wise by N steps using ARM TTDMI/LPC2148

Program:-

```
#include <LPC21xx.h>
void clock_wise (void);
void anti_clock_wise (void);
unsigned long int var1, var2;
unsigned int i=0, j=0, k=0;
int main (void)
{
PINSEL0 = 0x00FFFFFF;
IO0DIR = 0x0000F000;
while (1)
{
for (j=0; j<50; j++);
clock_wise ();
for (k=0; k<65000; k++);
for (j=0; j<50; j++);
anti_clock_wise ();
for (k=0; k<65000; k++);
}
}
```

```
void clock-wise (void)
{
var 1 = 0x00000800;
for (i=0; i<=3; i++)
{
var 1 = var 1 << 1;
var 2 = ~ var 1
var 2 = var 2 & 0x0000F000;
IOOPIN = ~ var 2;
for (K=0; K< 3000; K++);
}
}
void anti-clock-wise (void)
{
var 1 = 0x0001000;
for (i=0; i<=3; i++)
{
var 1 = var 1 >> 1;
var 2 = ~ var 1;
var 2 = var 2 & 0x0000F000;
IOOPIN = ~ var 2;
for (K=0; K< 3000; K++);
}
}
```

Q4. LED

```
#include <LPC21xx.h>
unsigned int delay;
int main()
{
PINSEL1 = 0x00000000;
IO0DIR = 0x00FF0000;
while (1)
{
IO0CLR = 0x00FF0000;
for (delay=0; delay<500000; delay++);
IO0SET = 0x00FF0000;
for (delay=0; delay<500000; delay++);
}
}
```

Q5. Binary Display code.

```
#include <LPC21XX.h>
void delay (void);
unsigned int hexvalue;
int main()
{
unsigned int not_hexvalue = 0;
PINSEL0 = 0x00000000;
IO0DIR = 0x00FF0000;
while(1)
{
for (hexvalue=0; hexvalue <= 0xff; hexvalue++)
{
not_hexvalue = (~hexvalue);
not_hexvalue &= 0x000000ff;
IO0PIN = (not_hexvalue << 16);
}
}
}
void delay (void)
{
unsigned int count.
for (count=0; count < 6500000; count++)
{}
}.
```

Shilpa