DSA Lab -1

Example 1:
Write a program to read n names of different sports and store them using array pointers. Use dynamic memory allocation and deallocation functions. The program should display all the names and deallocate the dynamic memory at the end of the program.

Code :

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main(){
int i,n;
char *sports[10];
char str[100];
printf("\n enter the number of sports \n");
scanf("%d", &n);
printf("\nenter the names of sports:\n");
for (i = 0; i < n; i++)
{
scanf("%s", str);
//allocating memory equal to the length of string + 1
//Last 1 byte to accommodate the '\0'
sports[i] = (char*) calloc(strlen(str)+1, sizeof(char));
strcpy(sports[i],str);
}
printf("\nGiven list of sports: \n");
for (i = 0; i < n; i++)
printf("%s\n", sports[i]);
//Deallocate the dynamic memory
for (i = 0; i < n; i++)
free(sports[i]);
return 0;
}
```

Output :

```
$ ls
a.out  week1_eg1.c
$ ./a.out

 enter the number of sports
5

enter the names of sports:
golf
hockey
polo
badminton
tennis

Given list of sports:
golf
hockey
polo
badminton
tennis
```

Example 2 :
Write a C program to implement a ragged array dynamically.

Code :

```
#include<stdio.h>
#include<stdlib.h>
int main(){
int rowNum, colNum, i, j;
int **table;
printf("\n enter the number of rows \n");
scanf("%d", &rowNum);
table = (int **) calloc(rowNum+1, sizeof(int *));
for (i = 0; i < rowNum; i++) /* this will tell which row we are in */
{
printf("enter size of %d row", i+1);
scanf("%d", &colNum);
table[i] = (int *) calloc(colNum+1, sizeof(int));
printf("\n enter %d row elements ", i+1);
for (j = 1; j <= colNum; j++)
{
scanf("%d", &table[i][j]);
}
table[i][0] = colNum;
printf("size of row number [%d] = %d", i+1, table[i][0]);
}
table[i] = NULL;
for (i = 0; i < rowNum; i++) /* this will tell which row we are in */
{
printf("displaying %d row elements\n", i+1);
for (j = 0; j <= *table[i]; j++)
{printf("%5d", table[i][j]);
}
printf("\n");
}
//freeup the memory
for (i = 0; i < rowNum; i++) {
free(table[i]);
}
free(table);
return 0;
}
```

Output :

```
$ gcc week1eg2.c
$ ./a.out

 enter the number of rows
5
enter size of 1 row2

 enter 1 row elements 1
2
size of row number [1] = 2enter size of 2 row3

 enter 2 row elements 3
4
5
size of row number [2] = 3enter size of 3 row4

 enter 3 row elements 6
7
8
9
size of row number [3] = 4enter size of 4 row5

 enter 4 row elements 10
11
12
13
14
size of row number [4] = 5enter size of 5 row6

 enter 5 row elements 15
16
17
18
19
20
size of row number [5] = 6displaying 1 row elements
    2    1    2
displaying 2 row elements
    3    3    4    5
displaying 3 row elements
    4    6    7    8    9
displaying 4 row elements
    5   10   11   12   13   14
displaying 5 row elements
    6   15   16   17   18   19   20
```

Questions for LAB -1

1) Write a function Smallest to find the smallest element in an array using pointer. Create a dynamically allocated array and read the values from keyboard in main. Display the result in the main function.

Code :

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

void Smallest(int n,int *ptr);

int main()
{
   int n,i,*ptr,*arr,small;
   printf("Enter the number of elements : ");
   scanf("%d",&n);

   arr = (int *)calloc (n,sizeof(int));
   if(arr==NULL)
   {
      printf("\nMemory not allocated.\n");
   }
   else
   {
      printf("\nMemory has been successfully allocated using calloc.\n\nPopulate the array : \n");
   }
   for(i=0; i<n; i++)
      {
            scanf("%d",&arr[i]);
      }
   ptr = arr;
   Smallest(n,ptr);
   small = *ptr;
   printf("\nThe smallest number in the array is %d.\n",small);
   return 0;
}

void Smallest(int n,int *ptr)
{
   int *p;
   p = ptr;
   for(int j=1; j<n; j++)
   {
      if(ptr[j]<*p)
      {
         *p = ptr[j];
      }
   }
   *ptr = *p ;
}
```

Output :

```
$ pwd
/home/Student/Desktop/200905130/DSA lab 1
$ ls
 a.out   bsl.h   dsa1.1.c  'running  C program with our own library.png'
$ gcc dsa1.1.c
$ ./a.out
Enter the number of elements : 5

Memory has been successfully allocated using calloc.

Populate the array :
5
0
-1
63
23

The smallest number in the array is -1.
```

2) Implement a C program to read, display and to find the product of two matrices using functions with suitable parameters. Note that the matrices should be created using dynamic memory allocation functions and the elements are accessed using array dereferencing.

Code :

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

int main()
{
   int i, j, m, n, p, q;
   int **a,**b,**c;
   printf("Enter dimension for a \n");
   scanf("%d %d",&m,&n);
   a = (int **) calloc(m,sizeof(int *));
   for(i=0; i<m; i++)
   {
      a[i] = (int *)calloc(n,sizeof(int));
   }
   printf("\nEnter dimension for b \n");
   scanf("%d %d", &p,&q);
   b = (int **) calloc(p,sizeof(int *));
   for(i=0; i<p; i++)
   {
      b[i] = (int *)calloc(q,sizeof(int));
   }

   if(a==NULL||b==NULL)
   {
```

```
    printf("Memory allocation failed.");
}

if(n!=p)
{
    printf("\nNOT MULTIPLICABLE\n");
    exit(0); // Terminate the execution
}
//else we can multiply the matrices and the order of the resultant matrix would be m x q
c = (int **) calloc(m,sizeof(int *));
for(i=0; i<m; i++)
{
    c[i] = (int *)calloc(q,sizeof(int));
}
printf("Memory has been successfully allocated");
printf("\nEnter elements for 1st matrix : \n");
for (i=0; i<m; i++)
{
    for(j=0; j<n; j++)
    {
        scanf("%d",&a[i][j]);
    }
}

printf("\nEnter elements for 2nd matrix : \n");

for(i=0; i<p; i++)
{
    for(j=0; j<q; j++)
    {
        scanf("%d",&b[i][j]);
    }
}

//multiplication
for(i=0; i<m; i++)
    for(j=0; j<q; j++)
    {
        *(*(c+i)+j) = 0;
        for(int k=0; k<n; k++)
        {
            *(*(c+i)+j) = *(*(c+i)+j) + *(*(a+i)+k) * *(*(b+k)+j);
        }
    }
printf("\nThe product matrix is : \n");
for(i=0; i<m; i++)
{
    for(j=0; j<q; j++)
    {
        printf("%d\t",*(*(c+i)+j));
    }
    printf("\n");
```

```
    }

    return 0;
}
```

Output :

```
$ pwd
/home/Student/Desktop/200905130/DSA lab 1
$ ls
 1.1.png   bsl.h     'running  C program with our own library.png'
 1_2.c     dsa1.1.c   sports
$ gcc 1_2.c
$ ./a.out
Enter dimension for a
2
2

Enter dimension for b
2
2
Memory has been successfully allocated
Enter elements for 1st matrix :
1
2
3
4

Enter elements for 2nd matrix :
4
3
2
1

The product matrix is :
8       5
20      13
```

3) Samuel wants to store the data of his employees, which includes the following fields: (i) Name of the employee (ii) Date of birth which is a collection of {day, month, year} (iii) Address which is a collection of {house number, zip code and state}. Write a 'C' program to read and display the data of N employees using pointers to array of structures.

Code :

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


typedef struct
{
   int date,month,year;
}DOB;
```

```
typedef struct
{
   int house_no;
   long zipcode;
   char state[20];
}ADRS;


typedef struct
{
   char name[20];
   DOB dob;
   ADRS address;
}EMPLOYEE;


int main()
{
   EMPLOYEE emp[10];
   EMPLOYEE *ptr = emp;

   int N;
   printf("Enter the number of employees : ");
   scanf("%d",&N);

   for(int i=0; i<N; i++)
   {
      printf("Enter the name of employee %d\n",i+1);
      scanf("%s",(ptr+i)->name);
      printf("Enter the date of birth of employee %d in date,month,year format\n",i+1);
      scanf("%d%d%d",&((ptr+i)->dob.date),&((ptr+i)->dob.month),&((ptr+i)->dob.year));
      printf("Enter the address of employee %d in house number, zipcode, state format\n",i+1);
      scanf("%d %ld %s",&((ptr+i)->address.house_no),&((ptr+i)->address.zipcode),(ptr+i)-
>address.state);
   }
   printf("\n\nThe employee details are : \n");

   for(int i=0; i<N; i++)
   {
      printf("\nThe name of employee %d is %s\n",i+1,(ptr+i)->name);
      printf("The date of birth of employee %d in date-month-year format is %d-%d-%d\n",i+1,
(ptr+i)->dob.date,(ptr+i)->dob.month,(ptr+i)->dob.year);
      printf("The address of employee %d : \nHouse number - %d \nZipcode - %ld \nState - %s\
n",i+1,((ptr+i)->address.house_no),((ptr+i)->address.zipcode),((ptr+i)->address.state));
   }
   return 0;
}
```

Output :

```
$ pwd
/home/Student/Desktop/200905130/DSA lab 1/lab programs
$ ls
1.1.c  1.2.c  1.3.c  a.out  bsl.h  sports  week1eg1.c  week1eg2.c  week2eg3.c
$ gcc 1.3.c -o struct
$ ./struct
Enter the number of employees : 2
Enter the name of employee 1
Agasthya
Enter the date of birth of employee 1 in date,month,year format
12
3
1993
Enter the address of employee 1 in house number, zipcode, state format
05
673
Karnataka
Enter the name of employee 2
Shashank
Enter the date of birth of employee 2 in date,month,year format
13
4
2000
Enter the address of employee 2 in house number, zipcode, state format
13
887
Goa


The employee details are :

The name of employee 1 is Agasthya
The date of birth of employee 1 in date-month-year format is 12-3-1993
The address of employee 1 :
House number - 5
Zipcode - 673
State - Karnataka

The name of employee 2 is Shashank
The date of birth of employee 2 in date-month-year format is 13-4-2000
The address of employee 2 :
House number - 13
Zipcode - 887
State - Goa
```