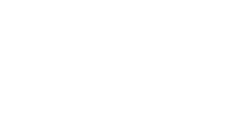


**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

**Batch: B2 Roll No.: 121** 

**Experiment / assignment / tutorial No. 9**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

| **TITLE:** Dynamic Memory Allocation. |
| --- |

**AIM:** Program to demonstrate dynamic memory allocation using malloc() & free () function.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Expected OUTCOME of Experiment:**

CO4: Design modular programs using functions and demonstrate the concept of pointers and file handling.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Books/ Journals/ Websites referred:**

1. Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.

2. Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill. 3. Introduction to programming and problem solving , G. Michael Schneider ,Wiley India edition.

4. **http://cse.iitkgp.ac.in/~rkumar/pds-vlab/**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Problem Definition:**

Write a C program to create a student list of a class using Dynamic memory allocation. It will have the details of students as roll number and name. Program should support the following operations (menu driven).

1. Insert

2. Delete

3. Display

use malloc for insert and free for delete

**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

**Algorithm:**

Step 1: Start

Step 2: Declare a structure named “school”

Step 2.1: Declare data members roll\_num and name, which are of integer and character data types, respectively. Name is a character array of size 100.

Step 3: Code in the main() function with void return type.

Step 4: Declare variables stu, ext\_stu, tot\_stu, i and ch – all integer data types. Step 5: Accept the value of “stu”, which is, the number of data to be entered.

Step 6: Declare a pointer “sch\_ptr” of data type, which is the user-defined data type struct school.

Step 7: Using malloc() function, dynamically allocate memory to the pointer variable sch\_ptr; sufficient memory to store “stu” number of data.

Step 8: If sch\_ptr = NULL,

Display appropriate error message and exit from the program.

Else

Step 8.1: For loop. Initialize i 🡨 0. Repeat the steps if i < stu. Increment the value of i by 1 after each iteration.

Step 8.1.1: Accept roll\_num and name using pointer operator.

Step 9: Display series of options for the user.

Step 10: Accept the choice “ch” from the user.

Step 11: Use Switch case.

Case 1:

Step 1: Accept the number of extra data to be added (ext\_stu)

Step 2: Declare an integer variable orig\_stu and initialize it as orig\_stu 🡨 stu Step 3: tot\_stu 🡨 stu + ext\_stu

Step 4: Reallocate memory dynamically using realloc() function; sufficient memory to store tot\_stu number of data.

**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

Step 5: If sch\_ptr = NULL

Display appropriate error message and exit from the program.

Else

For loop. Initialize i 🡨 orig\_stu. Repeat the loop if i < tot\_stu . Increment the value of i by one after each iteration.

Step 5.1: Accept roll\_num and name from the user.

Step 6: Display the final list: For loop. Initialize i 🡨 0. Repeat if i < tot\_stu. Increment the value of i by 1 after each iteration.

Step 6.1: Display the Sr. No., Roll No. and Name in a tabular format. Step 7: Break from case 1.

Case 2:

Step 1: Free the memory allocated dynamically to pointer variable sch\_ptr using the free() function.

Step 2: Break from case 2.

Case 3:

Step 1: Display the original list: For loop. Initialize i 🡨 0. Repeat if i < stu. Increment the value of i by 1 after each iteration.

Step 1.1: Display the Sr. No., Roll No. and Name in a tabular format. Step 7: Break from case 3.\

Default:

Step 1: Display the appropriate error message and exit from the program. Step 12: Stop

**Implementation details:**

#include<stdio.h>

#include<stdlib.h>

struct school

{

int roll\_num;

**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

char name[100];

};

void main()

{

int stu, ext\_stu, tot\_stu, i, ch;

printf("\nThis is the Students List of Class 12A.");

printf("\nEnter the number of student data to be entered: ");

scanf("%d", &stu);

struct school \*sch\_ptr;

sch\_ptr = (struct school\*)malloc(stu\*sizeof(struct school));

if(sch\_ptr==NULL){

printf("\nMemory allocation using Malloc was unsuccessful."); exit(0);

}

else{

printf("\nMemory allocation using Malloc was successful.");

for(i = 0; i < stu; i++)

{

printf("\nFor Sr. No. %d:", i+1);

printf("\nRoll No.: ");

scanf("%d", &(sch\_ptr+i)->roll\_num);

printf("\nName: ");

scanf("%s", (sch\_ptr+i)->name);

}

}

printf("\nEnter:\n'1' to insert student data\n'2' to delete the student data\n'3' to display the student list\nEnter choice: ");

scanf("%d", &ch);

switch(ch)

{

case 1:

printf("\nEnter the number of extra data to be added: ");

scanf("%d", &ext\_stu);

int orig\_stu = stu;

tot\_stu = stu + ext\_stu;

sch\_ptr = realloc(sch\_ptr, tot\_stu\*sizeof(struct school));

if(sch\_ptr==NULL){

printf("\nMemory allocation using Malloc was unsuccessful."); exit(0);

}

else{

printf("\nMemory allocation using Malloc was successful."); printf("\nNow enter the students' data: ");

for(i = orig\_stu; i < tot\_stu; i++)

{

printf("\nFor Sr. No. %d:", i+1);

**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

printf("\nRoll No.: ");

scanf("%d", &(sch\_ptr+i)->roll\_num);

printf("\nName: ");

scanf("%s", (sch\_ptr+i)->name);

}

}

printf("\nThe Final List is being displayed now: ");

printf("\nSr. No.\tRoll No.\tName\n");

for(i = 0; i < tot\_stu; i++)

printf("%d\t%d\t\t%s\n", i+1, (sch\_ptr+i)->roll\_num, (sch\_ptr+i)->name); break;

case 2:

free(sch\_ptr);

printf("\nThe Memory allocated to the pointer ''sch\_ptr'', which is\na pointer to the structure ''school'' containing data\nmembers ''roll\_num'' and ''name[100]'', has been successfully freed.\n");

exit(0);

break;

case 3:

printf("\nThe Original List is being displayed now: ");

printf("\nSr. No.\tRoll No.\tName\n");

for(i = 0; i < stu; i++)

printf("%d\t%d\t\t%s\n", i+1, (sch\_ptr+i)->roll\_num, (sch\_ptr+i)->name); break;

default:

printf("\nPlease enter either '1', '2' or '3' only and try again!"); exit(0);

}

}

**Output(s):**

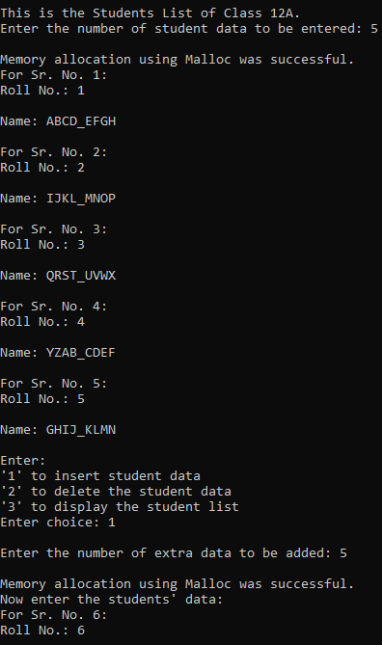
**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77**

(A Constituent College of Somaiya Vidyavihar University)



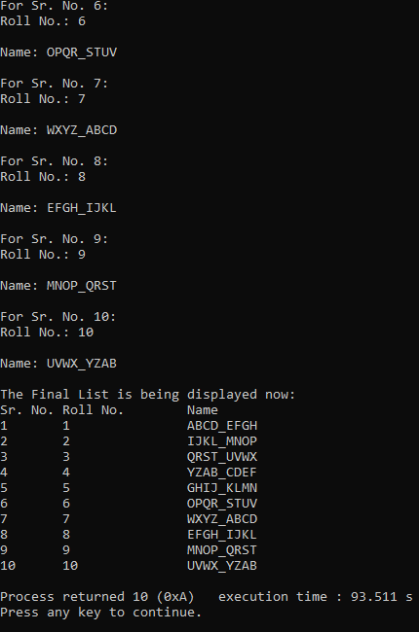
**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77**

(A Constituent College of Somaiya Vidyavihar University)



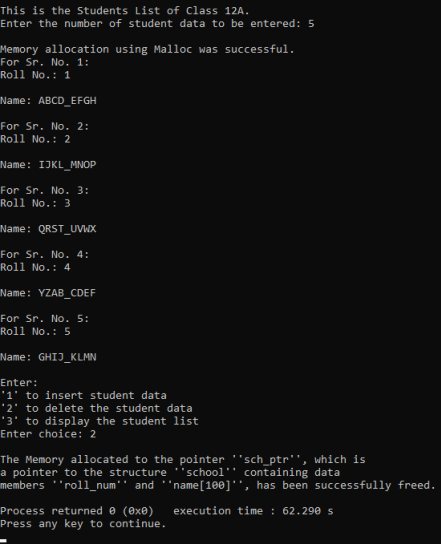
**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77**

(A Constituent College of Somaiya Vidyavihar University)



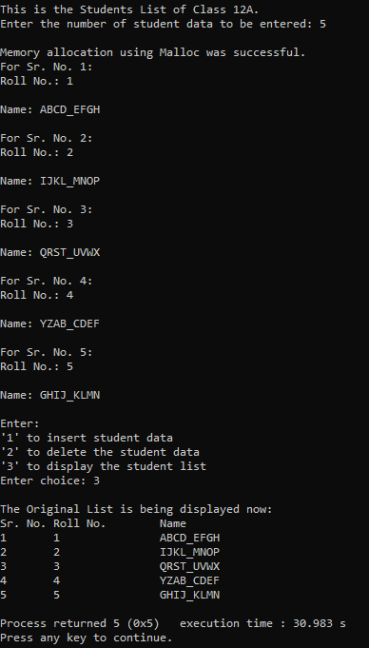
**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77**

(A Constituent College of Somaiya Vidyavihar University)



**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77**

(A Constituent College of Somaiya Vidyavihar University)



**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

**Conclusion:**

Thus, the concept of pointers and Dynamic Memory Allocation has been shown in this program. Pointers were used in Dynamic Memory Allocation in order to allocate memory space as per requirement during the running of the program. This has been achieved by the use of the malloc() function. Furthermore, during runtime, the memory requirement can again be changed with the help of the realloc() function. Thus, the limitations of Static Memory Allocation associated with arrays have been solved. This means that the chances of the program not running as expected is less and hence the program is more versatile and reliable.

**Post Lab Descriptive Questions**

1. **What is the difference between malloc and calloc?**

Ans.

| Sr. No. | malloc() | calloc() |
| --- | --- | --- |
| 1. | The memory block allocated is contiguous but the memory allocated is a single block of size equal to the value calculated from the input given  Ex., suppose that 5\*sizeof(int) has been entered inside malloc(). So a single memory block of 5×4 = 20 bytes is allocated. | The memory block allocated is contiguous but the memory allocated is divided into a number of equal-sized blocks, whose number equals the number entered in the function.  Ex. Suppose that 5, sizeof(int) has been entered inside calloc(). So 5 blocks of 4 bytes each are allocated contiguously. |
| 2. | It does not initialize the block and thus the block may have some garbage value. | It initializes each block with the default value i.e., 0. |
| 3. | It has one parameter or argument. | It has two parameters or arguments. |
| 4. | Syntax:  Ptr = (data\_type \*)malloc(n\*sizeof(data\_type)) | Syntax:  Ptr = (data\_type \*)calloc(n, sizeof(data\_type)) |

2. **Consider the following C code. What will be the output?**

# include<stdio.h>

# include<stdlib.h>

void fun(int \*a)

{

a = (int\*)malloc(sizeof(int));

}

int main()

{

int \*p;

fun(p);

\*p = 6;

printf("%d\n",\*p);

return(0);

**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021



**K. J. Somaiya College of Engineering, Mumbai-77** (A Constituent College of Somaiya Vidyavihar University)

}

(A) Compiler Error

(B) 6

(C) Runtime Error

(D) Garbage Value

Ans. (C) Runtime Error

3. **Difference between Static and Dynamic Memory allocation**

Ans.

| Sr. No. | Static Memory Allocation | Dynamic Memory Allocation |
| --- | --- | --- |
| 1. | In Static Memory Allocation, the variables get allocated permanently till the program ends or the function call ends. | In Dynamic Memory Allocation, the variables get allocated only when that program unit or function gets activated or called. |
| 2. | It is done before program execution. | It is done during program execution. |
| 3. | It uses stack for managing the static allocation of memory. | It uses heap for managing the dynamic allocation of memory. |
| 4. | It is less efficient. | It is more efficient. |
| 5. | There is no memory re-usability. | There is memory re-usability as allocated memory can be freed after the function’s call is over. This freed memory can be used again. |

**Date: 21-01-2022\_ Signature of faculty in-charge**

**Department of Science and Humanities**

Page No PIC Sem I/August-December 2021