Write a C Program using Dynamic Memory Allocation for the following problem statements.

1.to create memory for int, char and float variable at run time.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int \*iVar;

char \*cVar;

float \*fVar;

//allocating memory dynamically

iVar=(int\*)malloc(1\*sizeof(int));

cVar=(char\*)malloc(1\*sizeof(char));

fVar=(float\*)malloc(1\*sizeof(float));

printf("Enter integer value: ");

scanf("%d",iVar);

printf("Enter character value: ");

scanf(" %c",cVar);

printf("Enter float value: ");

scanf("%f",fVar);

printf("Inputted value are: %d, %c, %.2f\n",\*iVar,\*cVar,\*fVar);

//free allocated memory

free(iVar);

free(cVar);

free(fVar);

return 0;

}

2. to input and print text using Dynamic Memory Allocation.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int n;

char \*text;

printf("enter limit of the text:");

scanf("%d",&n);

//allocating memory dynamically

text=(char\*)malloc(1\*sizeof(char));

printf("Enter text: ");

scanf(" ");

gets(text);

printf("inputed text is:%s\n",text);

//free allocated memory

free(text);

return 0;

}

3. to read a one dimensional array, print sum of all elements along with  
inputted array elements using Dynamic Memory Allocation.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int \*arr;

int limit,i;

int sum=0;

printf("enter total number of element:");

scanf("%d",&limit);

//allocating memory dynamically

arr=(int\*)malloc(limit\*sizeof(int));

if(arr==NULL)

{

printf("insufficient memory\n");

return 0;

}

printf("enter %d elements:\n",limit);

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

{

printf("enter element %3d:",i+1);

scanf("%d",(arr+1));

sum=sum+ \*(arr+i);

}

printf("array elements are:");

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

printf("%3d",\*(arr+i));

printf("\nsum of all elements:%d\n",sum);

return 0;

}

4. to read and print the student details using structure and Dynamic Memory  
Allocation

#include <stdio.h>

#include <stdlib.h>

//structure declaration

struct student

{

char name[30];

int roll;

float perc;

};

int main()

{

struct student \*pstd;

pstd=(struct student\*)malloc(1\*sizeof(struct student));

if(pstd==NULL)

{

printf("insufficient memory\n");

return 0;

}

printf("Enter name:");

gets(pstd->name);

printf("Enter roll number:");

scanf("%d",&pstd->roll);

printf("Enter percentage:");

scanf("%f",&pstd->perc);

printf("\nEnter details are:\n");

printf("Name:%s,Roll number:%d,percentage:%.2f\n",pstd->name,pstd->roll,pstd->perc);

return 0;

}

5. to find sum of N elements entered by user. To perform this program,  
allocate memory dynamically using malloc() function.

#include <stdio.h>

#include <stdlib.h>

//structure declaration

struct student

{

char name[30];

int roll;

float perc;

};

int main()

{

struct student \*pstd;

int n,i;

printf("Enter total number of elements:");

scanf("%d",&n);

pstd=(struct student\*)malloc(n\*sizeof(struct student));

if(pstd==NULL)

{

printf("insufficient memory\n");

return 0;

}

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

{

printf("\nEnter details of student [%3d]:\n",i+1);

printf("Enter name:");

scanf(" ");

gets((pstd+i)->name);

printf("Enter roll number:");

scanf("%d",&(pstd+i)->roll);

printf("Enter percentage:");

scanf("%f",&(pstd+i)->perc);

}

printf("\nEntered details are:\n");

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

{

printf("%30s\t %5d\t %.2f\n",(pstd+i)->name,(pstd+i)->roll,(pstd+i)->perc);

}

return 0;

}

6. to find Largest of N Numbers. To perform this program, allocate memory  
dynamically using calloc() and realloc() function.

#include <stdlib.h>

int main() {

int num;

float \*data;

printf("Enter the total number of elements: ");

scanf("%d", &num);

// Allocating memory for num elements

data = (float \*)calloc(num, sizeof(float));

if (data == NULL) {

printf("Error! memory not allocated.");

exit(0);

}

// Storing numbers entered by the user.

for (int i = 0; i < num; ++i) {

printf("Enter Number %d: ", i + 1);

scanf("%f", data + i);

}

// Finding the largest number

for (int i = 1; i < num; ++i) {

if (\*data < \*(data + i))

\*data = \*(data + i);

}

printf("Largest number = %.2f", \*data);

return 0;

}