24 January, 2017

Codes from LAB-2, CSE135, Section: E.

**Problem 1**:

Write a C program to find sum and average of ‘n’ elements entered by user. To perform this program, allocate memory dynamically using malloc() function.

//Code for Problem 1.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int \*ptr, sum=0, i, n;

double ave;

printf("Enter how many numbers you want to add:\n");

scanf("%d",&n);

ptr = (int\*)malloc(n\*sizeof(int));

if(ptr==NULL)

{

printf("Not allocated\n");

exit (0);

}

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

{

scanf("%d",ptr+i);

sum+=\*(ptr+i);

}

ave = (double)sum/n;

printf("Sum = %d\nAverage = %0.2lf",sum,ave);

free(ptr);

return 0;

}

**Problem 2**:

Write a C program to find first ‘n’ odd numbers entered by user. To perform this program, allocate memory dynamically using malloc() function.

//Code for Problem 2

#include <stdio.h>

#include <stdlib.h>

int main()

{

int \*p, n, i, j=1;

printf("Enter length of the series:\n");

scanf("%d",&n);

p = (int\*)malloc(n\*sizeof(int));

if(p==NULL)

{

printf("Not allocated\n");

exit (0);

}

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

{

\*(p+i) = j;

j+=2;

}

printf("Series :");

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

{

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

}

free(p);

return 0;

}

**Problem 3:**

Write a C program to dynamically allocate memory for ‘n’ integers with malloc() function and print the addresses of the allocated memory units.

//Code for Problem 3

#include <stdio.h>

#include <stdlib.h>

main()

{

int \*p, n, i;

printf("Enter the number of integers:\n");

scanf("%d",&n);

p = (int\*)malloc(n\*sizeof(int));

if(p==NULL)

{

printf("Not allocated\n");

exit (0);

}

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

printf("Address of unit %d is %p\n",i+1,p+i);

free(p);

return 0;

}

**Problem 4:**

Write a C program using malloc() to print a series for ‘n’ as given below;

When n = 3, Series: 4 12 20

When n = 4, Series: 4 12 20 28

//Code for Problem 4

#include <stdio.h>

#include <stdlib.h>

main()

{

int \*ptr, n, i, j=4;

printf("Enter length of the series:\n");

scanf("%d",&n);

ptr = (int\*)calloc(n,sizeof(int));

if(ptr==NULL)

{

printf("Not allocated\n");

exit (0);

}

for(i=0;i<4;i++)

{

\*(ptr+i)=j;

j+=8;

}

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

{

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

}

free(ptr);

return 0;

}

**Problem 5:**

Write a C program to input ‘n’ numbers using calloc() and linearly search a number from the input. If the searched number is in the list, print ”Found”, the memory address and the location unit for the given input. And if not found, simply print “Not Found”.

//Code for Problem 5

#include <stdio.h>

#include <stdlib.h>

main()

{

int \*ptr, n, i, s, truth=0;

printf("Enter the number of input:\n");

scanf("%d",&n);

ptr = (int\*)calloc(n,sizeof(int));

if(ptr==NULL)

{

printf("Not allocated\n");

exit (0);

}

printf("Enter values:\n");

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

scanf("%d",&\*(ptr+i));

printf("Enter a digit to search: ");

scanf("%d",&s);

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

{

if(\*(ptr+i)==s)

{

truth++;

break;

}

}

if(truth==0)

printf("Not Found\n");

else

printf("Found at address %p, block %d\n",ptr+i,i+1);

}

**Problem 6:**

Write a C program to allocate memory dynamically for ‘n’ integers given by user using malloc() function and print the memory address of the allocated memory units. Now reallocate the memory again for different number of integers using realloc() function.

//Code for Problem 6

#include <stdio.h>

#include <stdlib.h>

main()

{

int \*ptr, n, n2, n3, i;

printf("Enter the number of integers for which you want to allocate memory:\n");

scanf("%d",&n);

ptr = (int\*)malloc(n\*sizeof(int));

if(ptr==NULL)

{

printf("Not allocated\n");

exit (0);

}

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

{

printf("Address %p\n",ptr+i);

}

printf("Enter the new number of integers:\n");

scanf("%d",&n2);

ptr = realloc(ptr, n2);

for(i=0;i<n2;i++)

{

printf("Address %p\n",ptr+i);

}

printf("Enter the new new number of integers:\n");

scanf("%d",&n3);

ptr = realloc(ptr, n3);

for(i=0;i<n3;i++)

{

printf("Address %p\n",ptr+i);

}

return 0;

}

**Problem 7:**

Write a C program to create a structure named ‘node’ with members ‘int x’ and ‘int y’. Now create an object of mentioned structure in main function using malloc(). At last, assign some values to the structure members and print them out.

//Code for Problem 7

#include <stdio.h>

#include <stdlib.h>

struct node

{

int x;

int y;

};

main()

{

struct node \*ptr = (struct node\*)malloc(sizeof(struct node));

ptr->x=5;

ptr->y=6;

printf("%d %d",ptr->x,ptr->y);

free(ptr);

}

-End-