Task1: Run and understand the following example.

#include <stdio.h>

#include <string.h>

int main(void) {

char t[20] = "ABCDEFGHIJK";

int s = strlen(t);

t[3] = '\0';

s += strlen(t);

strcpy(t,"ABCDEF");

s += strlen(t);

strcat(t,"ABC");

printf("%s \n",t);

s += strlen(t);

printf("%d \n",s);

return 0;

}

Task 2: Run and understand the following example.

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \* ptr;

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

if(ptr != NULL)

{

\*ptr=200;

printf("The pointer value is %d \n", \*ptr);

}

else {

printf("Memory allocation is not successful");

}

return 0;

}

Task 3: Run and understand the following example.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int\* ptr;

int n, i, sum = 0;

printf("Enter number of elements:");

scanf("%d", &n);

ptr = (int\*)malloc( n \* sizeof(int)); // points to beginning of block

printf("Address of first block: %d \n", ptr);

if (ptr == NULL) {

printf("Memory not allocated.\n");

exit(0);

}

else {

printf("Memory successfully allocated using malloc.\n");

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

\*(ptr+i) = i + 1;

//ptr[i]=i+1;

printf("Address of %d element is: %d \n" , i , ptr+i);

}

printf("The elements of the array are: ");

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

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

//printf("%d, ", ptr[i]);

}

}

return 0;

}

Task 4: Use calloc() instead of malloc in Task 3.

Task 5: Modify the task 3 so you can extend the array with as many elements as the user wishes.

Use realloc().

Task 6: Run and understand the following example.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int \*arr;

// malloc() allocate the memory for 5 integers

// containing garbage values

arr = (int \*)malloc(5 \* sizeof(int)); // 5\*4bytes = 20 bytes

// Deallocates memory previously allocated by malloc() function

free( arr );

// calloc() allocate the memory for 5 integers and

// set 0 to all of them

arr = (int \*)calloc(5, sizeof(int));

// Deallocates memory previously allocated by calloc() function

free(arr);

return(0);

}

Task 7:

Scenario

Write a program that allocates memory of a size requested by the user. This program requests a number from the user and checks if that number is less than 1 MB (1024\*1024). If it's not, then the program prints the message. "Too much memory requested.".

In the next step, the program allocates a one-dimesional array of characters (char) and fills this array with characters from "A" to "Z" (the 1st element (index 0) contains "A", the 26th element(index 25) contains "Z", the 27th element (index 26) contains "A" and so on). Then, the program prints the first 400 bytes of an array (or the whole array if it's smaller than 400 characters), 40 characters in each row. Remember to free up the array memory at the end of the program.

Your version of the program must print the same result as the expected output.

**Hint:** please use memory allocation, for loop and modulo operator.

Example input

100

Example output

ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMN

OPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZAB

CDEFGHIJKLMNOPQRSTUV

Example input

500

Example output

ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMN

OPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZAB

CDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOP

QRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZABCD

EFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQR

STUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZABCDEF

GHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRST

UVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGH

IJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUV

WXYZABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJ

Example input

1200500

Example output

Too much memory requested.

Homework will be uploaded to canvas and netacad after this class. The deadline is the 19th of November. Please send it through netacad like the previous homework.