WEEK 1 ASSESMENT

1. Write a C program to determine the given number is odd or even using Bitwise operators.

#include <stdio.h>

int main() {

int num;

printf("Enter an integer: ");

scanf("%d", &num);

if (num & 1) {

printf("%d is odd.\n", num);

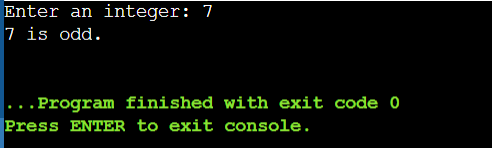
} else {

printf("%d is even.\n", num);

}

return 0;

}



2. Write a C program to count the number of bits set in a number.

Input:

144

Output:

Count of Set bits: 2

#include <stdio.h>

int cntSet(int num) {

int c = 0;

while (num > 0) {

c += num & 1;

num >>= 1;

}

return c;

}

int main() {

int num, res;

printf("Enter an integer: ");

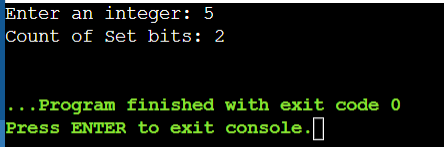
scanf("%d", &num);

res = cntSet(num);

printf("Count of Set bits: %d\n", res);

return 0;

}



3. Write a C program to swap two numbers. Use a function pointer to do this operation.

Input:

84 25

Output:

25 84

#include <stdio.h>

void swap(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int main() {

int num1, num2;

printf("Enter two numbers: ");

scanf("%d %d", &num1, &num2);

void (\*swapPtr)(int \*, int \*);

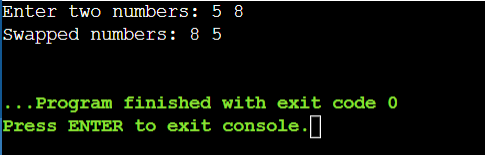
swapPtr = &swap;

swapPtr(&num1, &num2);

printf("Swapped numbers: %d %d\n", num1, num2);

return 0;

}



4. Write an equivalent pointer expression for fetching the value of array element a[i][j][k][2]

\*(\*(\*(\*(a + i) + j) + k) + 2)

5. Write a C program to Multiply two matrix (n\*n) using pointers.

Input:

Output:

Size of Row: 3

Product:

Size of Column: 3

48 39 30

Matrix 1:

102 84 66

2 3 4

129 111 93

5 6 7

8 9 1

Matrix 2:

9 8 7

6 5 4

3 2 1

#include <stdio.h>

#define MAX\_SIZE 10

void multiplyMatrices(int size, int (\*matrix1)[MAX\_SIZE], int (\*matrix2)[MAX\_SIZE], int (\*result)[MAX\_SIZE]) {

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

for (int j = 0; j < size; j++) {

\*(\*(result + i) + j) = 0;

for (int k = 0; k < size; k++) {

\*(\*(result + i) + j) += \*(\*(matrix1 + i) + k) \* \*(\*(matrix2 + k) + j);

}

}

}

}

void displayMatrix(int size, int (\*matrix)[MAX\_SIZE]) {

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

for (int j = 0; j < size; j++) {

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

}

printf("\n");

}

}

int main() {

int size;

int matrix1[MAX\_SIZE][MAX\_SIZE], matrix2[MAX\_SIZE][MAX\_SIZE], result[MAX\_SIZE][MAX\_SIZE];

printf("Size of Row: ");

scanf("%d", &size);

printf("Size of Column: ");

scanf("%d", &size);

printf("Matrix 1:\n");

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

for (int j = 0; j < size; j++) {

scanf("%d", &matrix1[i][j]);

}

}

printf("Matrix 2:\n");

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

for (int j = 0; j < size; j++) {

scanf("%d", &matrix2[i][j]);

}

}

multiplyMatrices(size, matrix1, matrix2, result);

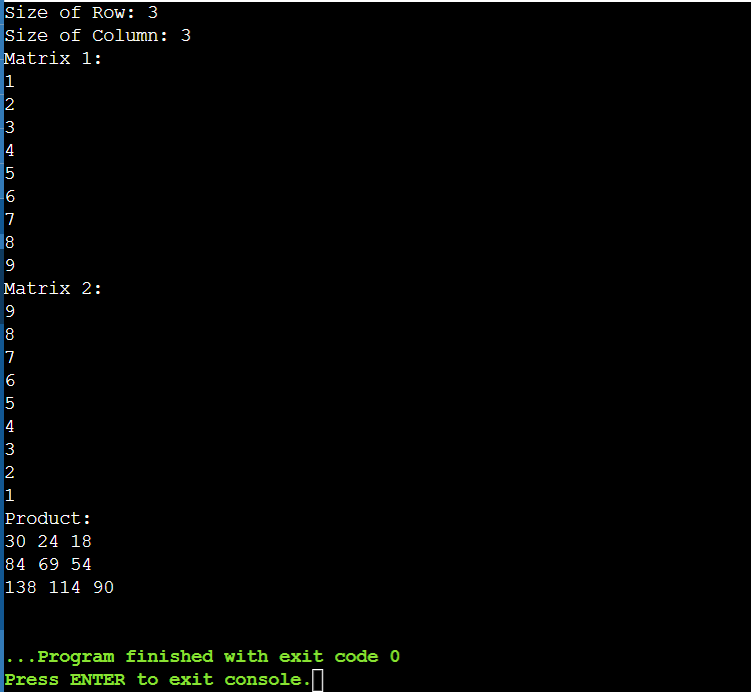
printf("Product:\n");

displayMatrix(size, result);

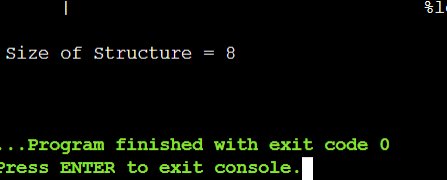
return 0;

}

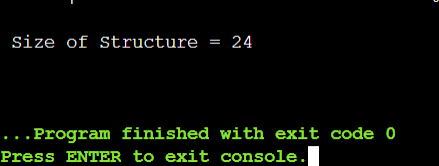
}



6. Find the output of the following // Consider the compiler is 32-bit machine



7. Find the output of the following // Consider the compiler is 32-bit machine



8. Find the output of the following // Consider the compiler is 32-bit machine

