**Lab No.6**

01. Write a program which will find the factorial of a given number. Exit the program if the input number is negative. Test case: Input number = 5,Factorial is=5\*4\*3\*2\*1

#include <stdio.h>

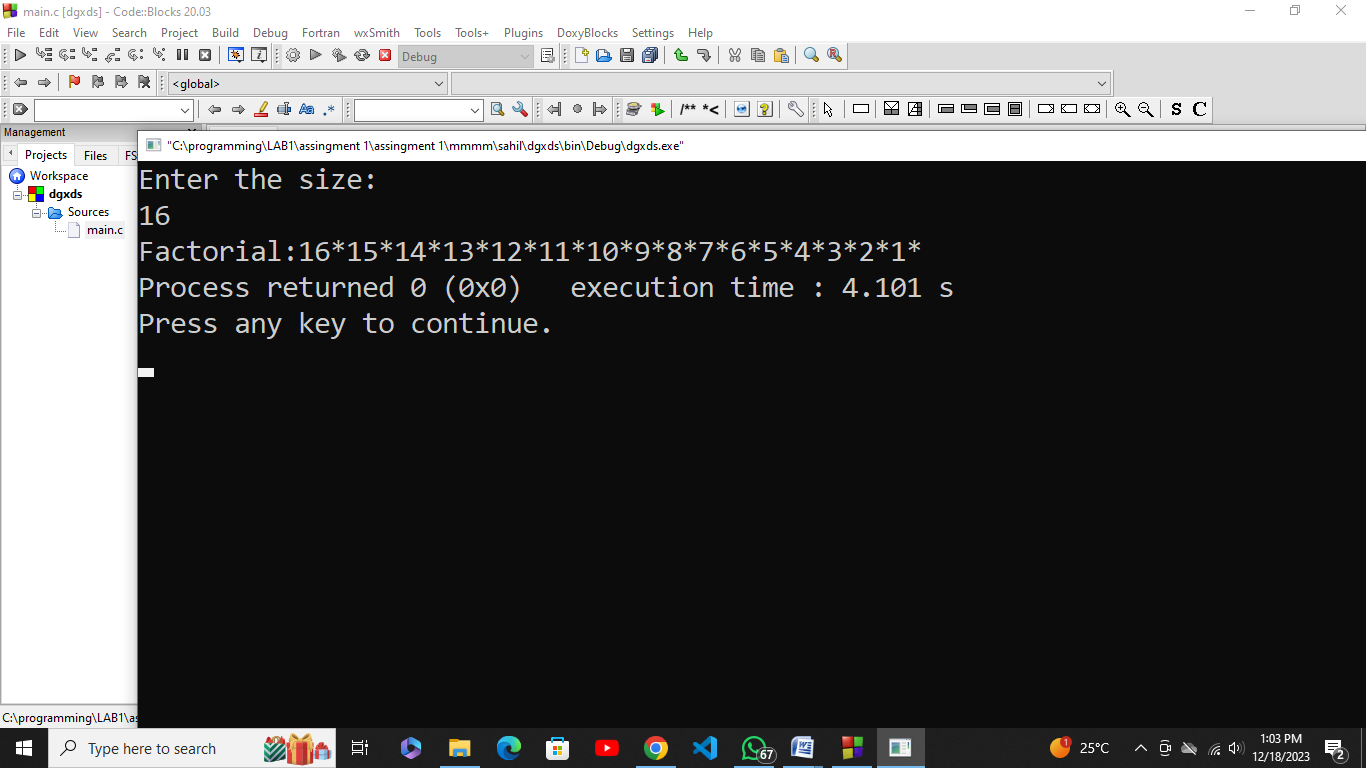
int main() {

int n;

printf("Enter the size:\n"); **Out put:**

scanf("%d",&n);

printf("Factorial:");



for(int i=n;i>=1;i--){

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

}

return 0;

}

02. Write a program which will generate the Fibonacci series up to 1000. Also find the sum of the generated Fibonacci numbers divisible by 3, 5 and 7 only. Fibonacci series is:1 1 2 3 5 8 13 25..........

Note: Do this task by using for loop and while loop. Also identify which one is more efficient?

#include <stdio.h>

#include <stdlib.h>

int main()

{

int digit1 = 1, digit2 = 1, x, add = 0;

printf("Using for loop method:\n");

for (int i = 0; digit1 <= 1000; ++i)

{

printf("%d ", digit1);

if (digit1 % 3 == 0 || digit1 % 5 == 0 || digit1 % 7 == 0)

{

add += digit1;

}

x = digit1 + digit2;

digit1 = digit2;

digit2 = x;

}

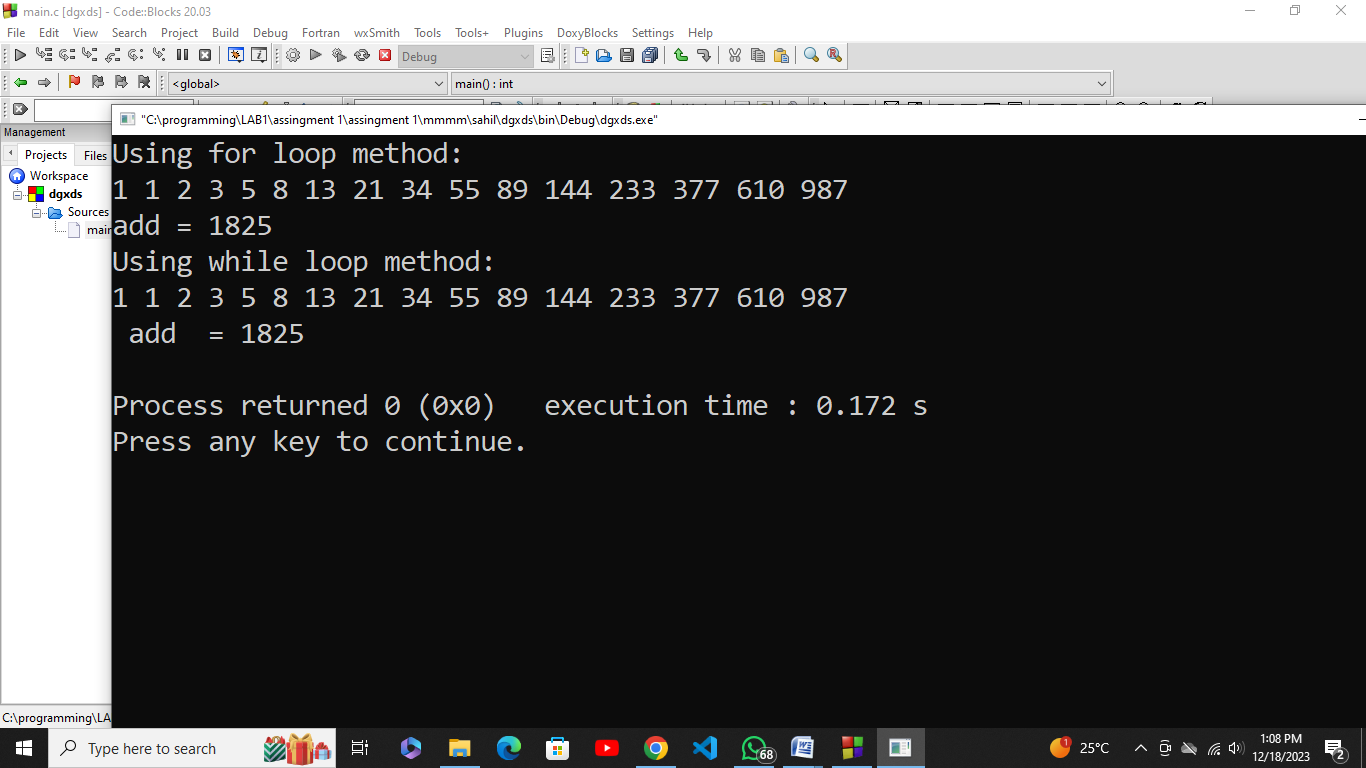
printf("\nadd = %d\n", add);

digit1 = 1;

digit2 = 1;

add= 0; **Out put:**

printf("Using while loop method:\n");



while (digit1 <= 1000)

{

printf("%d ", digit1);

if (digit1 % 3 == 0 || digit1 % 5 == 0 || digit1 % 7 == 0)

{

add += digit1;

}

x = digit1 + digit2;

digit1 = digit2;

digit2 = x;

}

printf("\n add = %d\n", add);

return 0;

}

03. Write a program which will input a 5-digit number. If the sum of digits is even, find whether the input number is a prime or not. If the sum of digits is odd find, whether the number is palindrome or not?

Example of prime number: A number which is only divisible by itself and 1 i.e., 7, 11, and13.

Example of a Palindrome: A number whose reverse order is the same as the original number i.e., 11211, 44344.

#include <stdio.h>

#include <stdlib.h>

int prime(int a)

{

if (a <= 1) {

return 0;

}

for (int i = 2; i \* i <= a; i++)

{

if (a % i == 0) {

return 0;

}

}

return 1;

}

int pal(int a)

{

int rev = 0;

int x = a;

while (x > 0) {

rev = rev \* 10 + x % 10;

x /= 10;

}

return a== rev;

}

int main()

{

int digit[5], add = 0;

for (int i = 0; i < 5; i++){

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

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

}

for (int i = 0; i < 5; i++) {

add += digit[i];

}

if (add % 2 == 0) {

printf("The add is an even number.\n");

if (prime(add)) {

printf("The add %d is a prime number.\n", add);

}else {

printf("The add% d is not a prime number.\n", add);

}

}else{

printf("The add is an odd number.\n");

if (pal(add)){

printf("The add %d is a palindrome.\n", add);

}else{

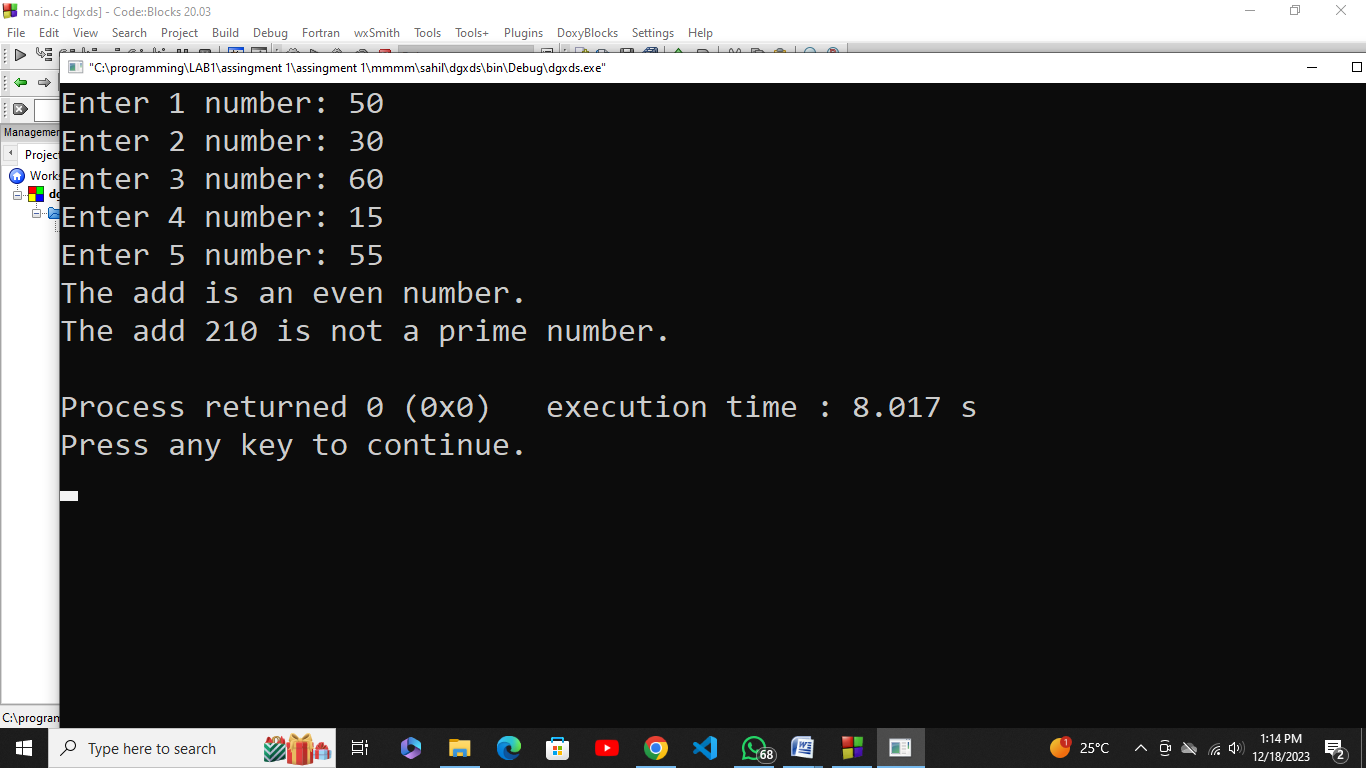
printf("The add %d is not a palindrome.\n", add);

}

} Out put:

return 0;

}



04. Develop a user-registration system have the following options.

1. Ask the user for a user-name (5 alphabets).
2. Password should be 6 characters long with at least 1 numeric, 1 capital and 1 small letter.
3. Display a “Account Created Successfully”.
4. Login the user with correct username and password.
5. Display “Welcome username, you are now logged in”.

**Note:** Develop your application using break and continue statement.

#include <stdio.h>

#include <stdlib.h>

int main(){

char username[6];

char password[7];

int i, hasNumeric, hasCapital, hasSmall;

while (1) {

printf("Enter username (5 alphabets): ");

scanf("%s", username);

for (i = 0; i < 6; i++) {

if (!username[i]){

break;

}

}

if (i != 5){

printf("Username should be 5 alphabets.\n");

continue;

}

printf("Enter password (6 characters with at least 1 numeric, 1 capital, and 1 small letter): ");

scanf("%s", password);

for (i = 0; i < 7; i++){

if (!password[i]){

break;

}

}

if (i != 6){

printf("Password should be 6 characters long.\n");

continue;

}

hasNumeric = hasCapital = hasSmall = 0;

for (i = 0; i < 6; i++){

if (password[i] >= '0' && password[i] <= '9'){

hasNumeric = 1;

}

else if (password[i] >= 'A' && password[i] <= 'Z'){

hasCapital = 1;

}

else if (password[i] >= 'a' && password[i] <= 'z'){

hasSmall = 1;

}

}

if (hasNumeric != 1 || hasCapital != 1 || hasSmall != 1){

printf("Password should contain at least 1 numeric, 1 capital, and 1 small letter.\n");

continue;

}

printf("Account Created Successfully.\n");

break;

}

while (1){

char inputUsername[6];

char inputPassword[7];

printf("Enter username for login: ");

scanf("%s", inputUsername);

for (i = 0; i < 6; i++){

if (!inputUsername[i]){

break;

}

}

if (i != 5){

printf("Invalid username. Try again.\n");

continue;

}

printf("Enter password for login: ");

scanf("%s", inputPassword);

for (i = 0; i < 7; i++){

if (!inputPassword[i]){

break;

}

}

if (i != 6){

printf("Invalid password. Try again.\n");

continue;

}

int usernameMatch = 1;

int passwordMatch = 1;

for (i = 0; i < 5; i++){

if (username[i] != inputUsername[i]){

usernameMatch = 0;

break;

}

}

for (i = 0; i < 6; i++) {

if (password[i] != inputPassword[i]){

passwordMatch = 0;

break;

}

}

if (usernameMatch && passwordMatch){

printf("Welcome %s, you are now logged in.\n", username);

break;

}

else{

printf("Invalid username or password. Try again.\n");

continue;

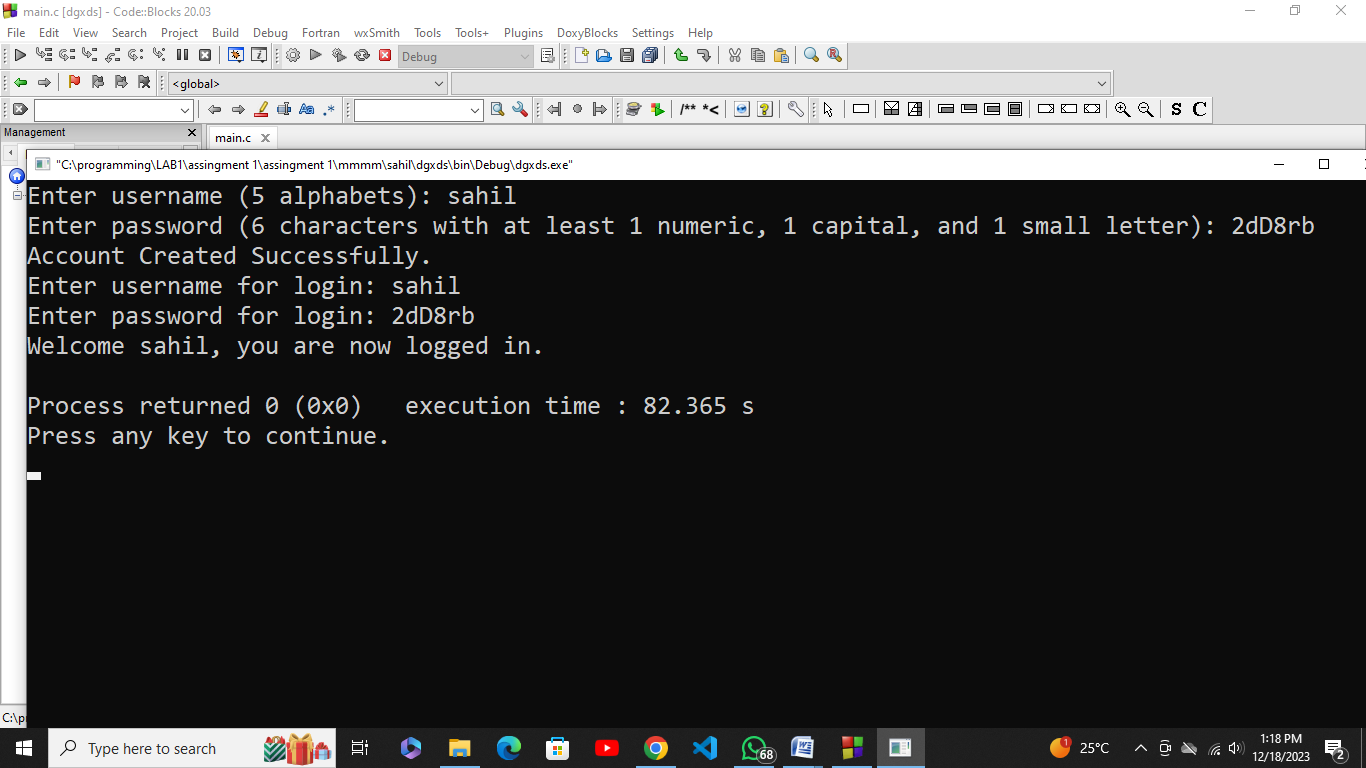
}

}

return 0;

}

**Out put:**



05. (Calculating the Value of π) Calculate the value of π from the infinite series. Print a table that shows the value of π approximated by one term of this series, by two terms, by three terms, and so on. How many terms of this series do you have to use before you num1 get 3.14?  
3.141? 3.1415? 3.14159?

(Calculating the Value of π) Calculate the value of π from the infinite series. Print a table that shows the value of π approximated by one term of this series, by two terms, by three terms, and so on. How many terms of this series do you have to use before you num1 get 3.14?  
3.141? 3.1415? 3.14159?

#include <stdio.h>

#include <stdlib.h>

int main()

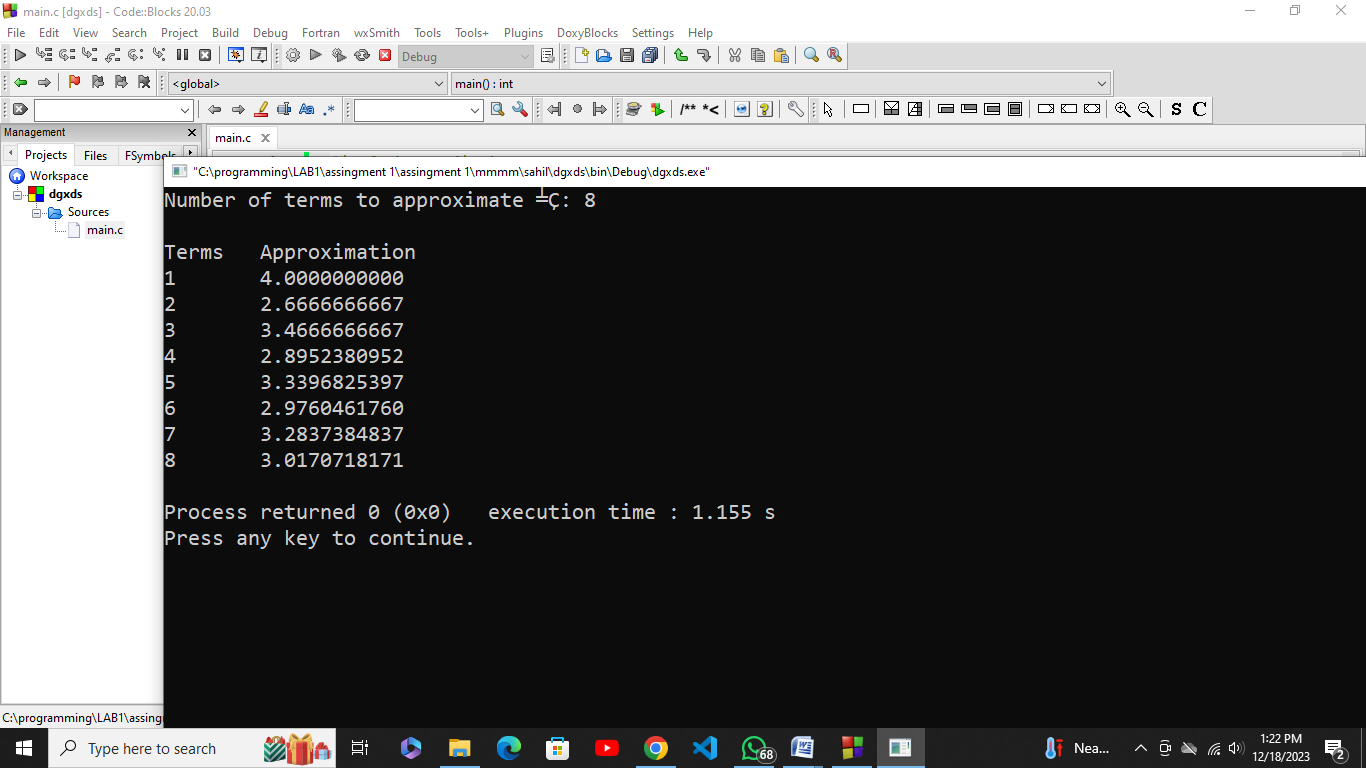
{

int numTerms;

double piApproximation = 0.0; **Out put:**

printf("Number of terms to approximate π: ");

scanf("%d", &numTerms);



printf("\nTerms\tApproximation\n");

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

if (i % 2 == 1){

piApproximation += 4.0 / (2 \* i - 1);

}else{

piApproximation -= 4.0 / (2 \* i - 1);

}

printf("%d\t%.10f\n", i, piApproximation);

if (piApproximation >= 3.14 && piApproximation <= 3.14159){

break;

}

}

return 0;

}

06. (Diamond-Printing Program) Write a program that prints the following diamond shape. You may use printf statements that print either a single asterisk (\*) or a single blank. Maximize your use of repetition (with nested for statements) and minimize the number of printf statements.

#include <stdio.h>

#include <stdlib.h>

int main() {

int i, j, k;

for (i = 0; i < 5; i++) {

for (j = 0; j < 5 - i; j++) {

printf(" ");

}

for (k = 0; k < 2 \* i + 1; k++) {

printf("\*");

}

printf("\n");

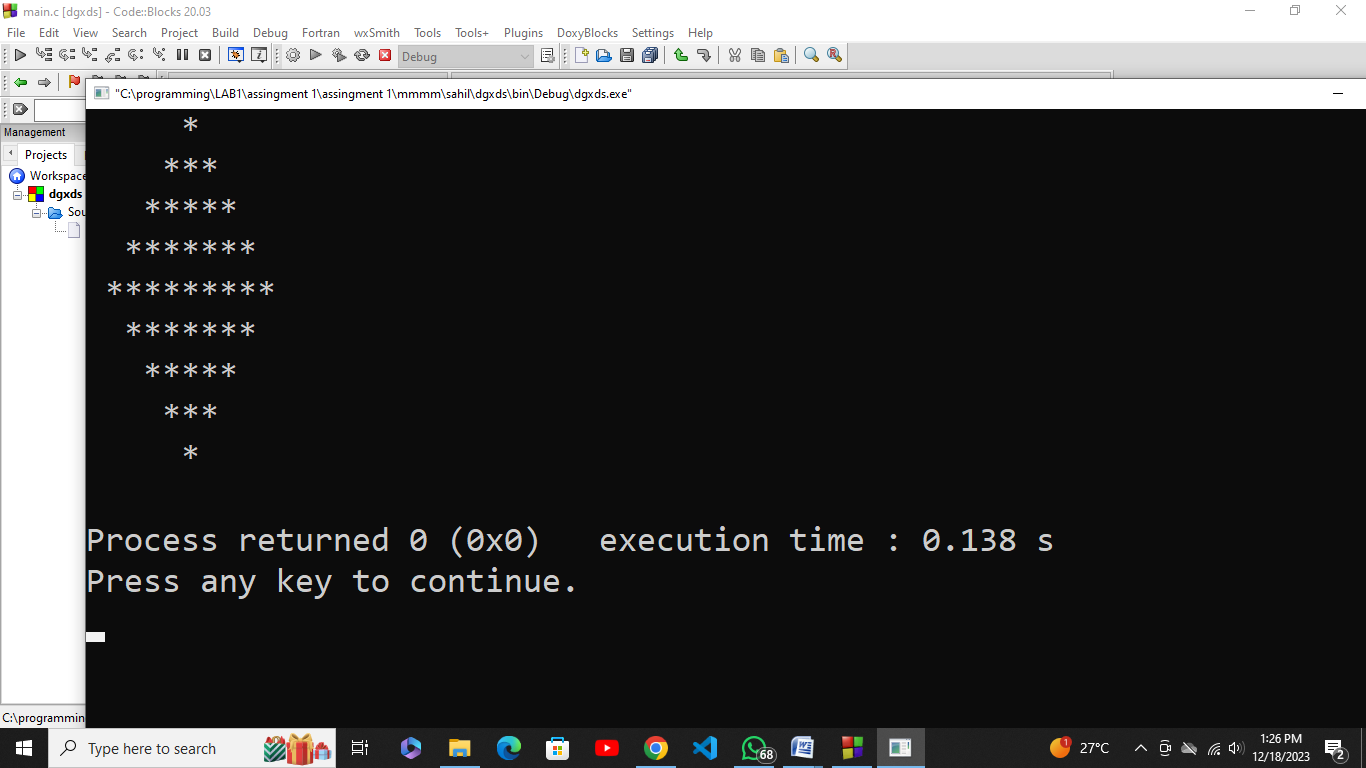
}

for (i = 5 - 2; i >= 0; i--) { **Out put:**

for (j = 0; j < 5 - i; j++) {

printf(" ");

}



for (k = 0; k < 2 \* i + 1; k++) {

printf("\*");

}

printf("\n");

}

return 0;

}