KPIT Technologies weekly assignment, (NHCE)

Week -2

Name :M.Sai Charishma,

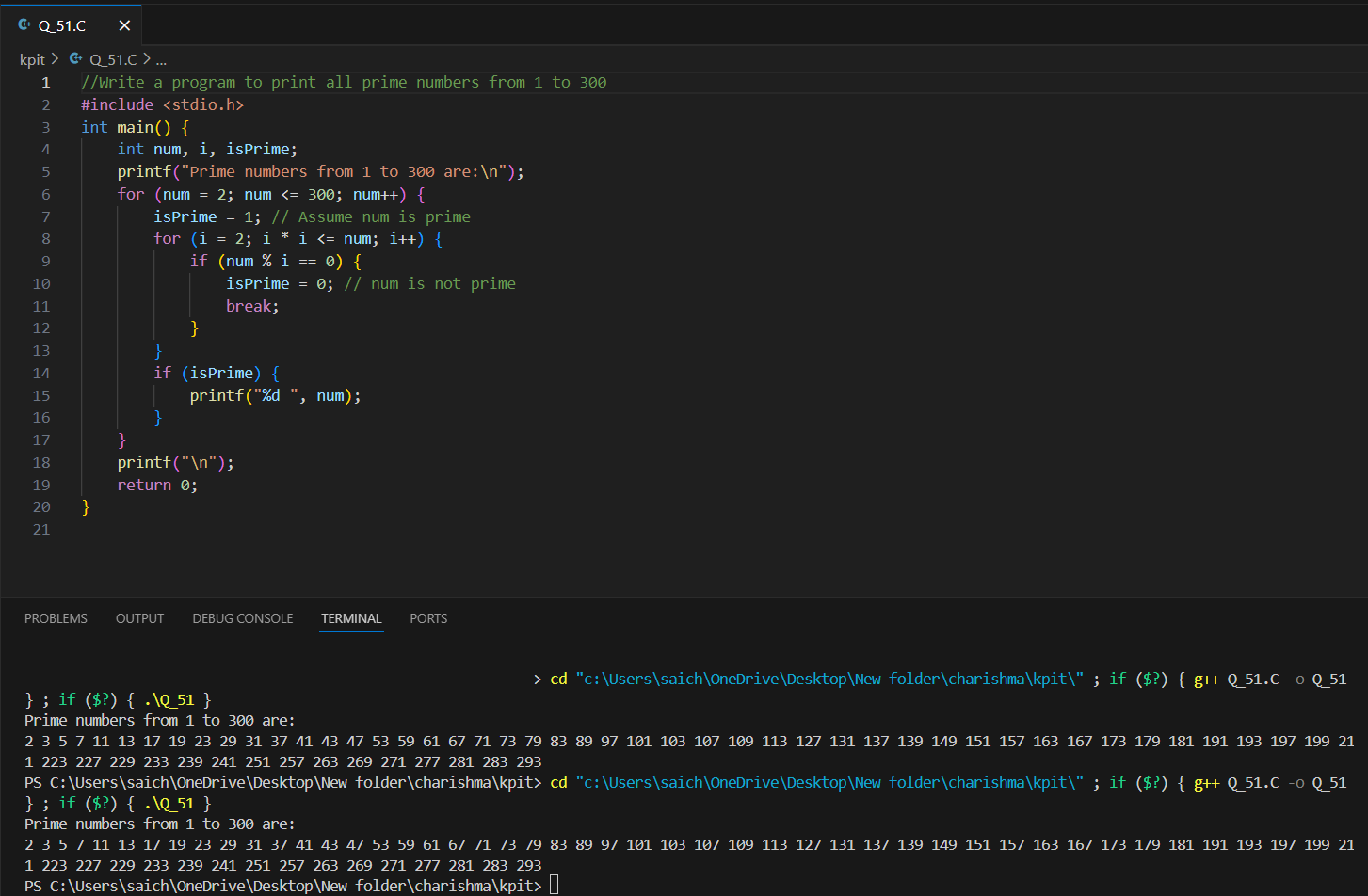
USN :1NH20EC077.

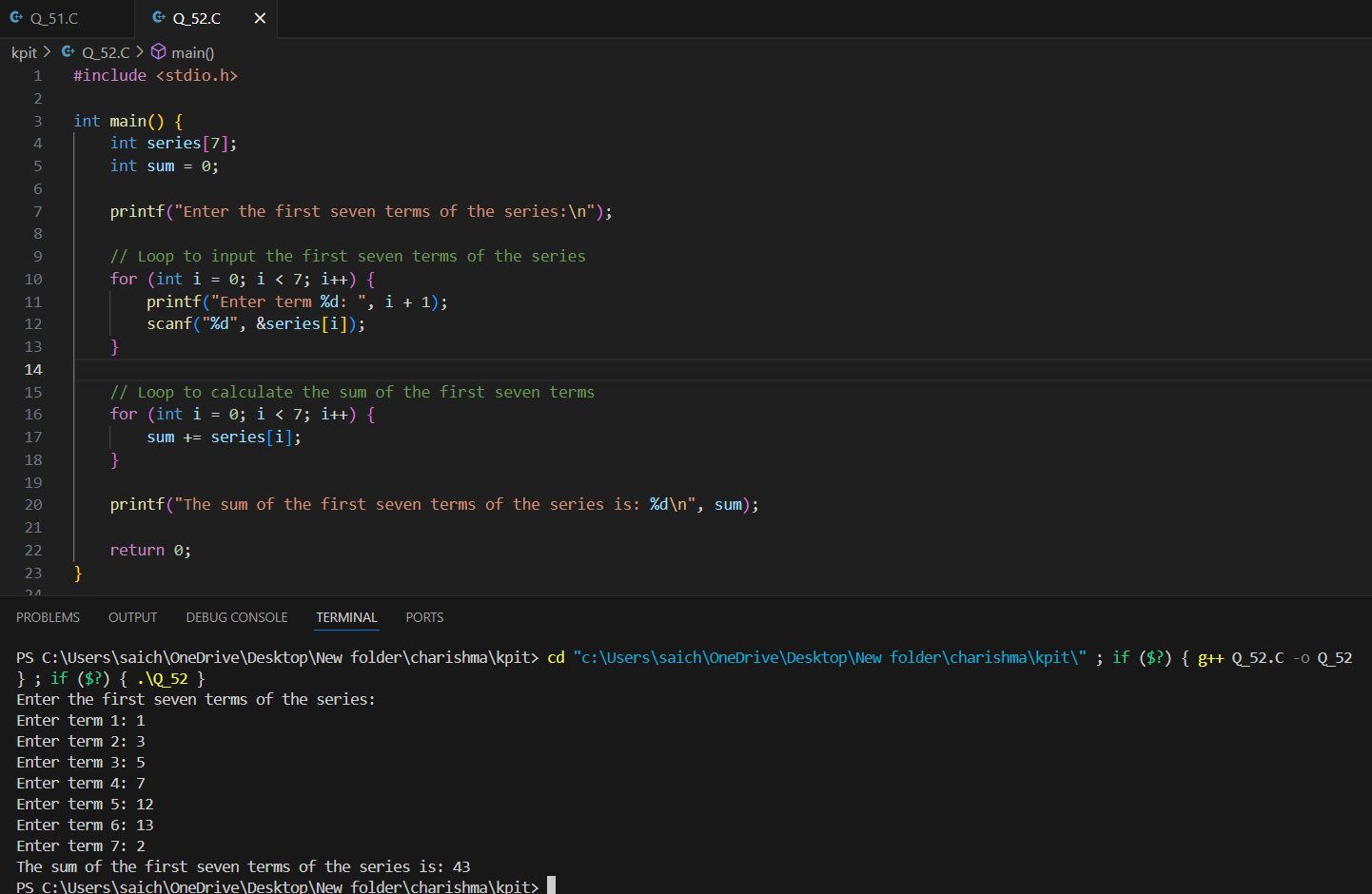
Branch : Circuit

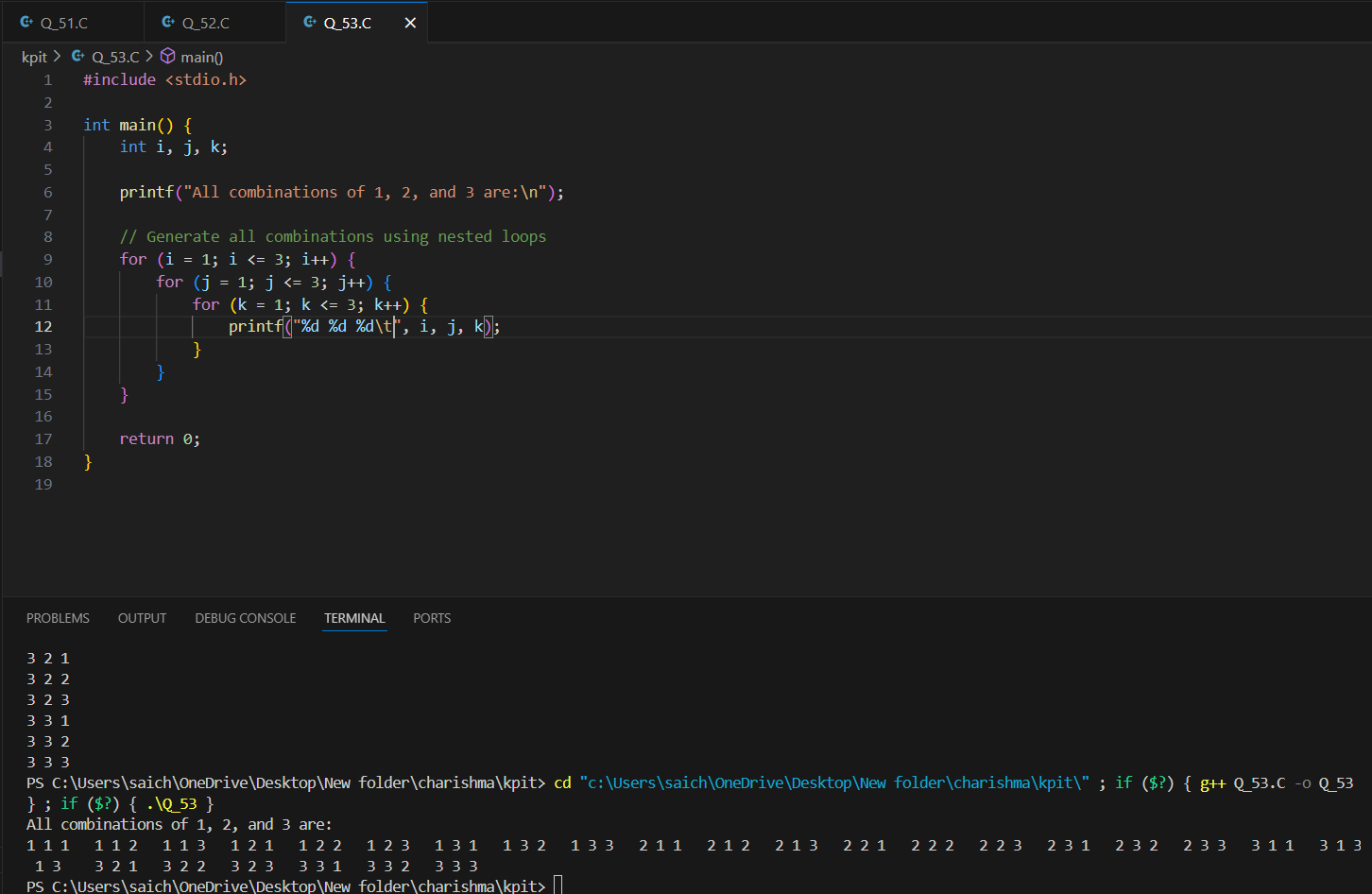
Peer Evaluation, Yashavant Kanetkar Book

Book : Let us C , 51- 100 questions

Date : 31/05/2024







**54 (a) The break statement is used to exit from:**

**1. An if statement**

**2. A for loop**

**3. A program**

**4. The main( ) function**

**(b) A do-while loop is useful when we want that the statements withinthe loop must be executed:**

**1. Only once**

**2. At least once**

**3. More than once**

**4. None of the above**

**(c) In what sequence the initialization, testing and execution of body is**

**done in a do-while loop?**

**1. Initialization, execution of body, testing**

**2. Execution of body, initialization, testing**

**3. Initialization, testing, execution of body**

**4. None of the above**

**(d)Which of the following is not an infinite loop?**

**1. int i = 1;**

**while (1)**

**{**

**i++;**

**}**

**2. for (; ;);**

**3. int t = 0, f;**

**while (t)**

**{**

**f = 1;**

**4. int y, x = 0;**

**do**

**{**

**y = x;**

**(e) Which of the following statements is true for the following**

**program?**

**# include <stdio.h>**

**int main()**

**{**

**int x = 10, y = 100 % 90;**

**for (i = 1; i <= 10; i++);**

**if ( x != y ) ;**

**printf ( "x = %d y = %d\n", x, y ) ;**

**return 0 ;**

**}**

**1. The printf( ) function is called 10 times.**

**2. The program will produce the output x = 10 y = 10.**

**3. The ; after the if ( x != y ) will not produce an error.**

**4. The program will not produce any output.**

**5. The printf( ) function is called infinite times.**

**(f) Which of the following statement is true about a for loop used in a**

**C program?**

**1. for loop works faster than a while loop.**

**2. All things that can be done using a for loop can also be done**

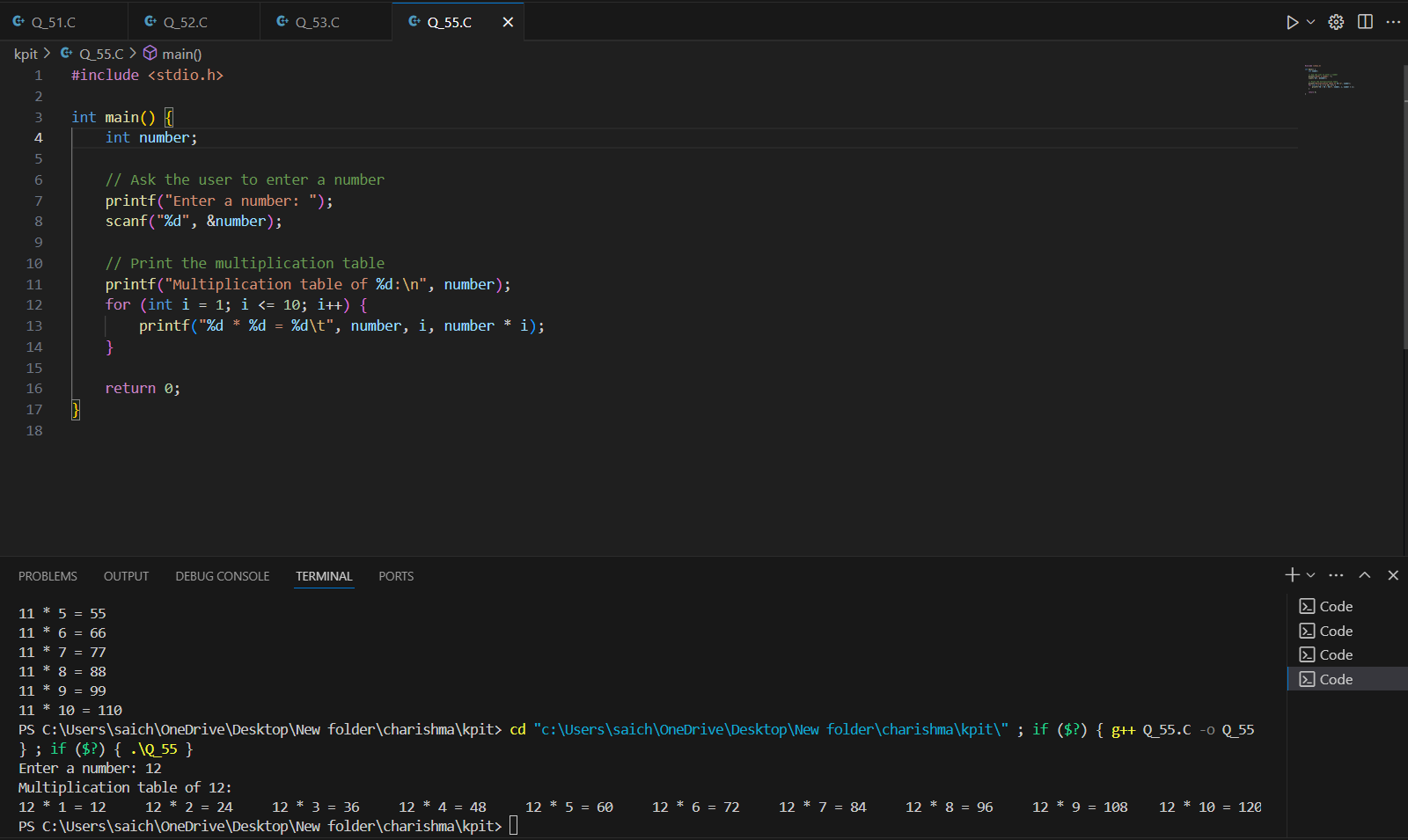
**using a while loop.**

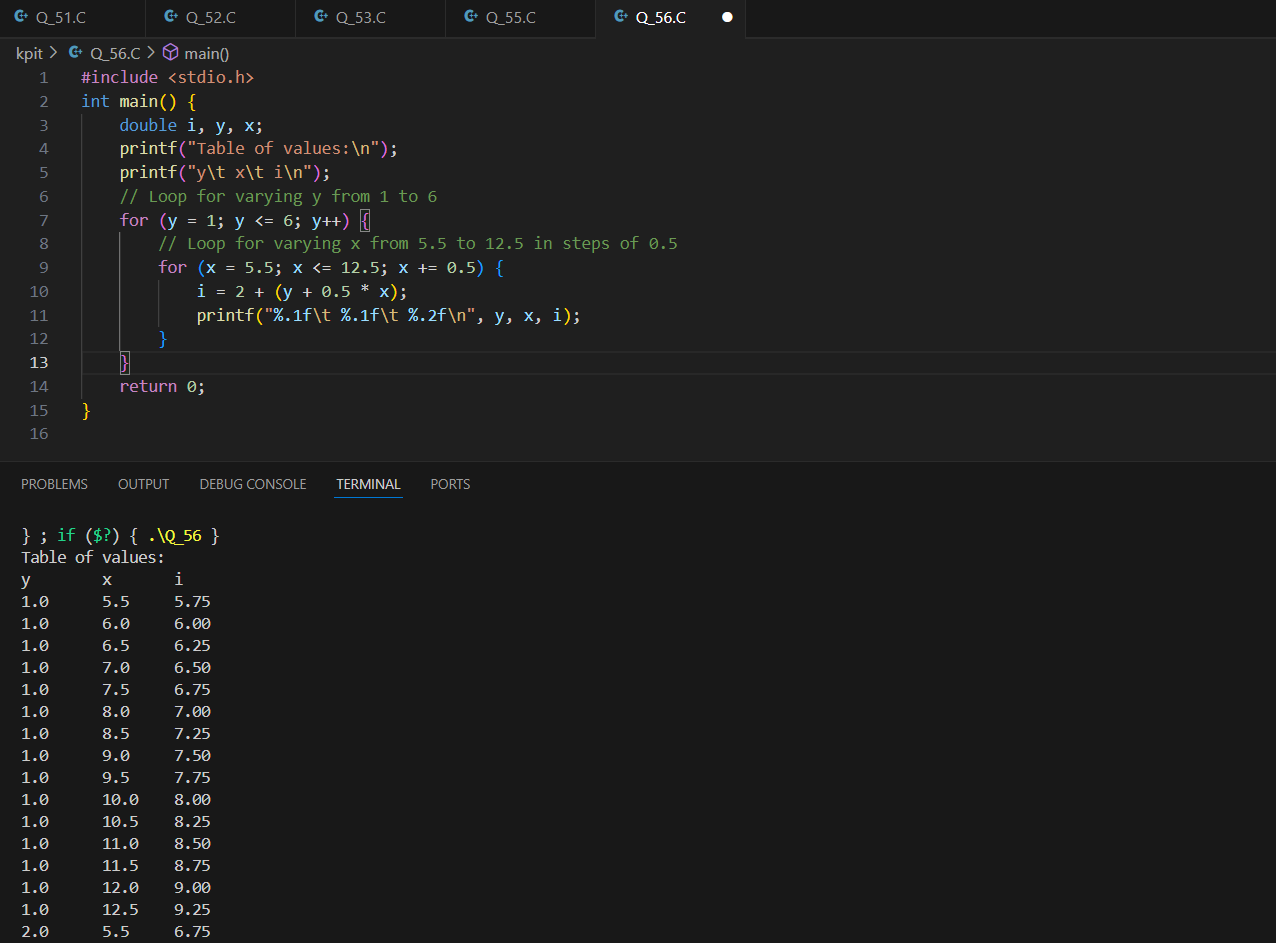
**3. for ( ; ; ) implements an infinite loop.**

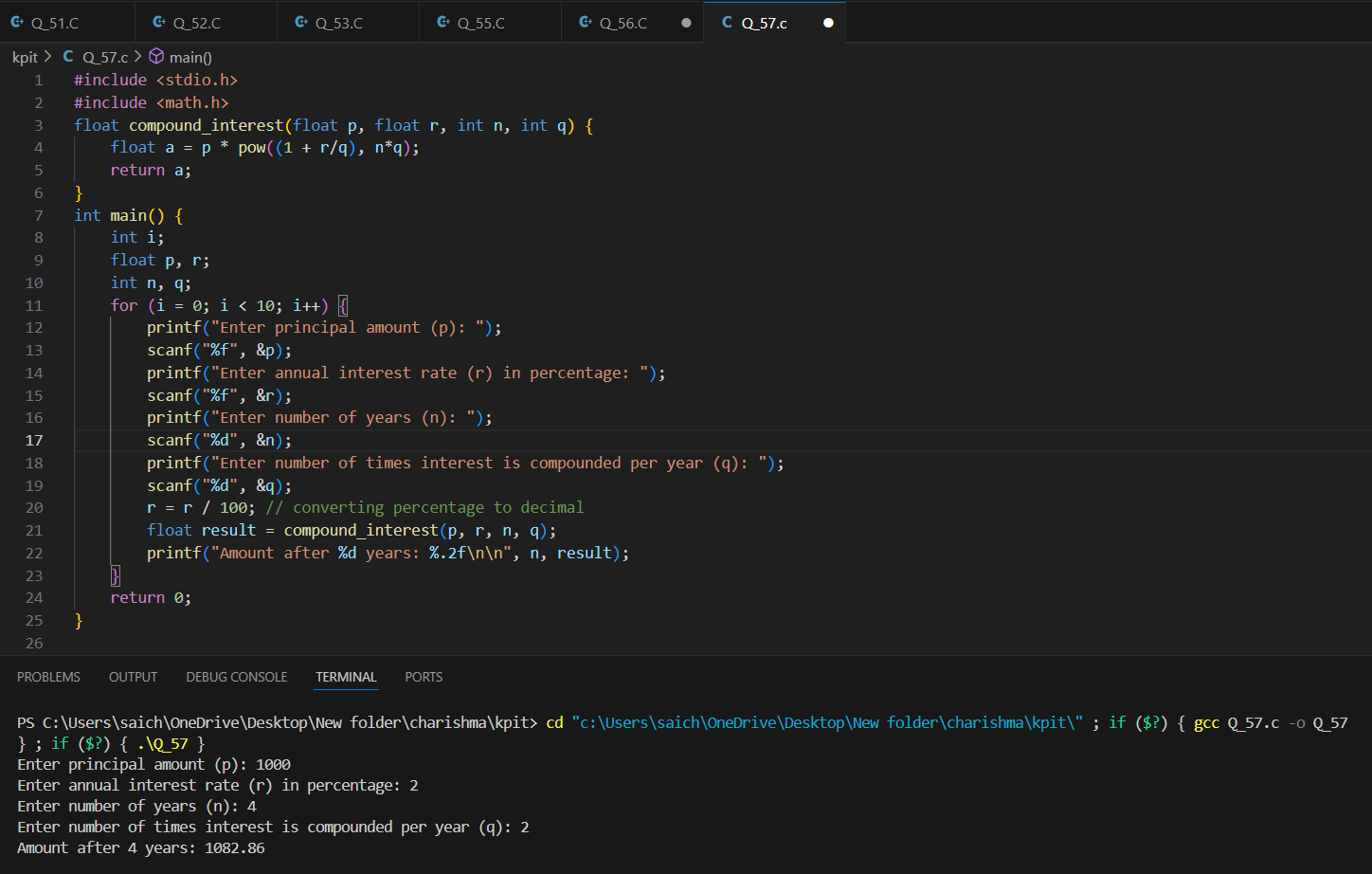
**4. for loop can be used if we want statements in a loop to get**

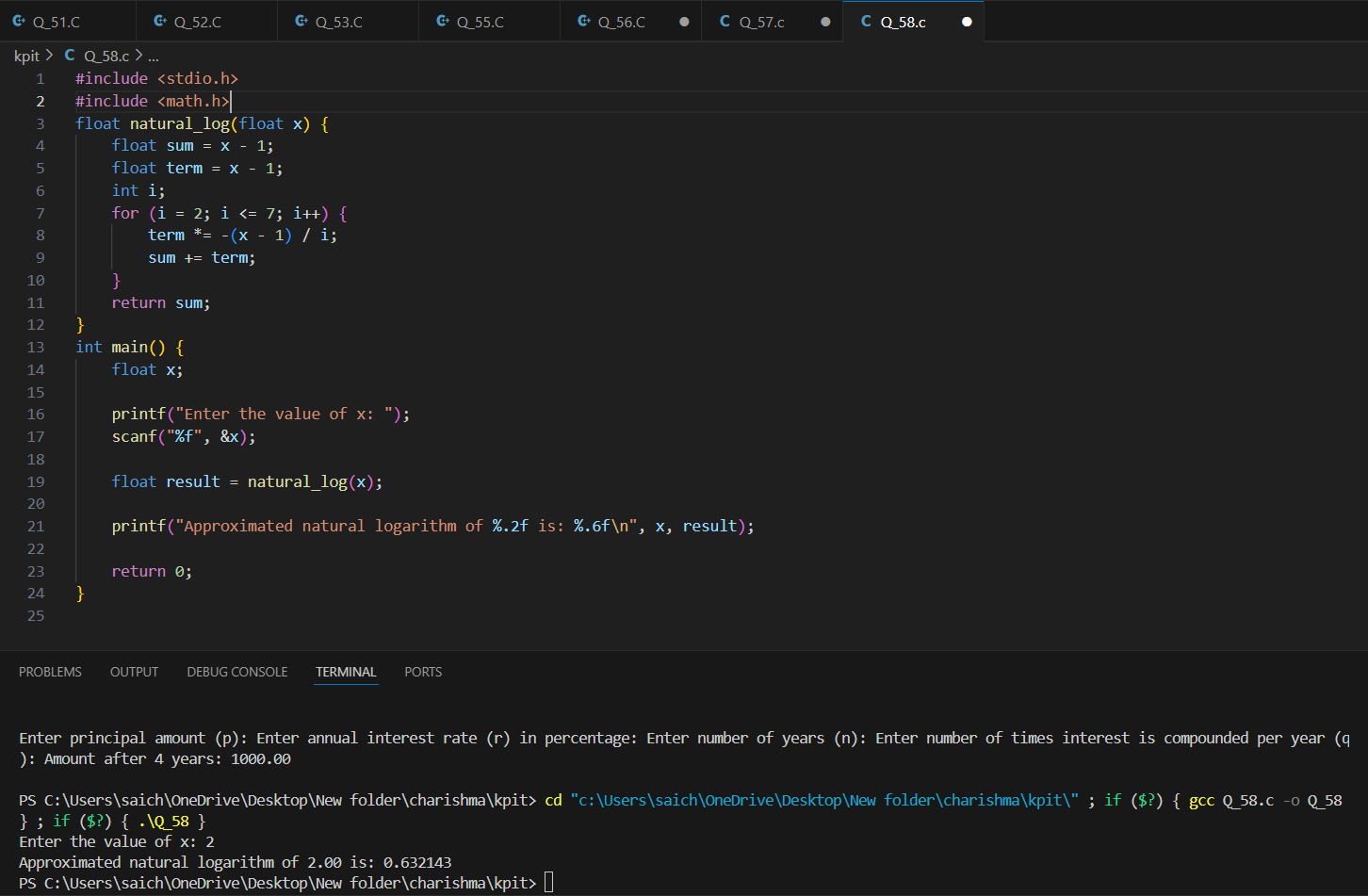
**executed at least once.**

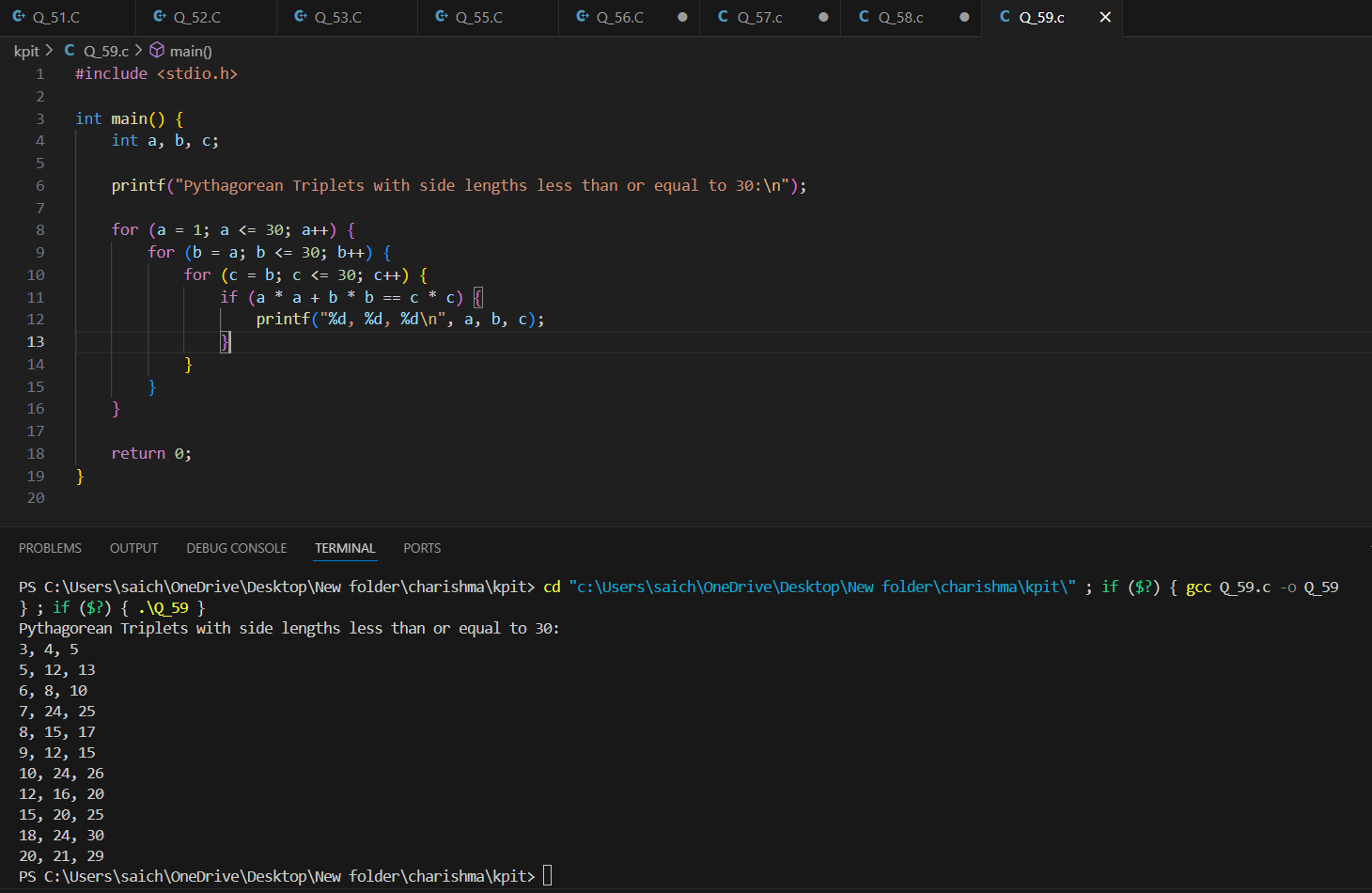
**5. for loop works faster than a do-while loop.**

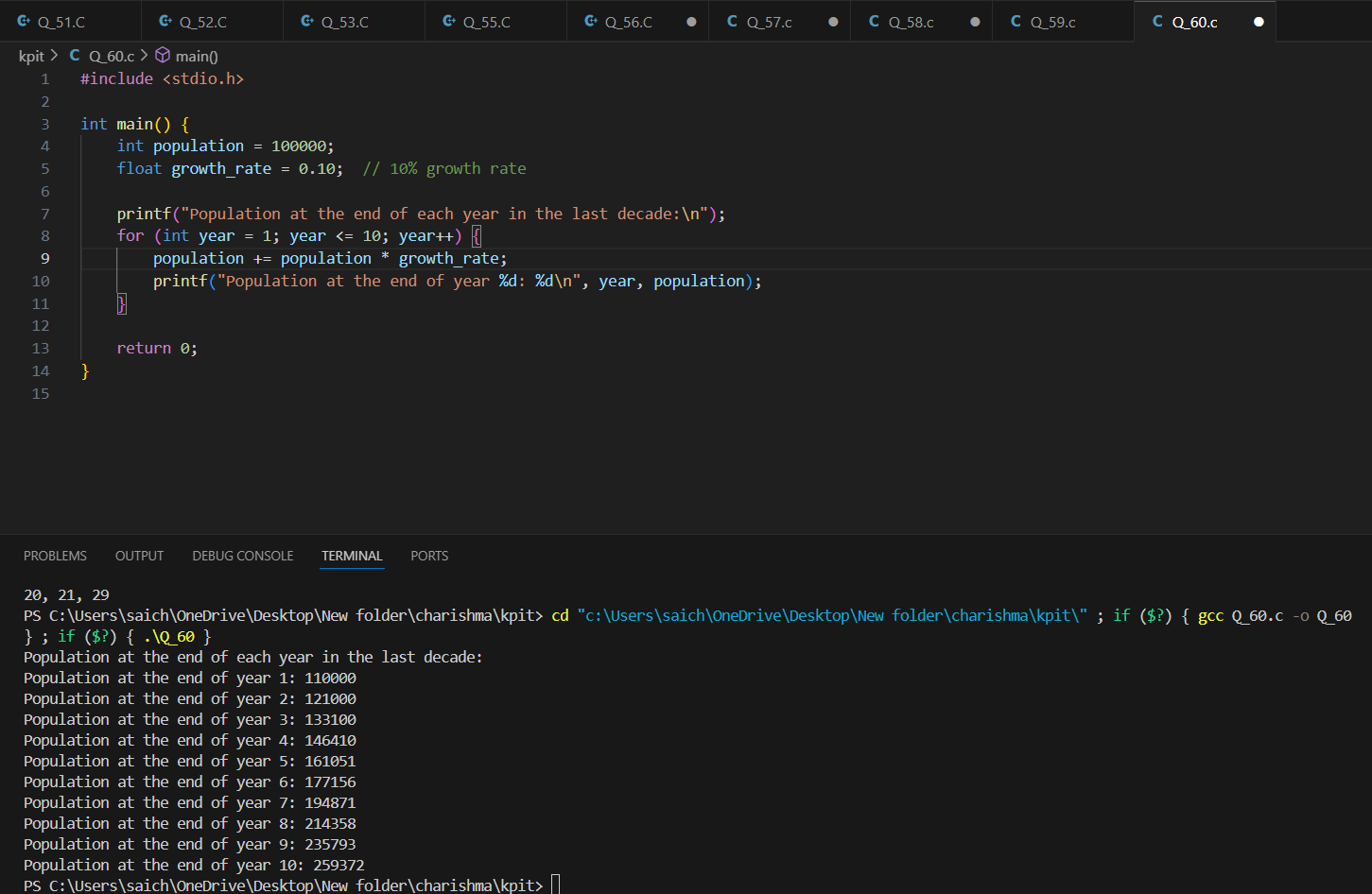
****

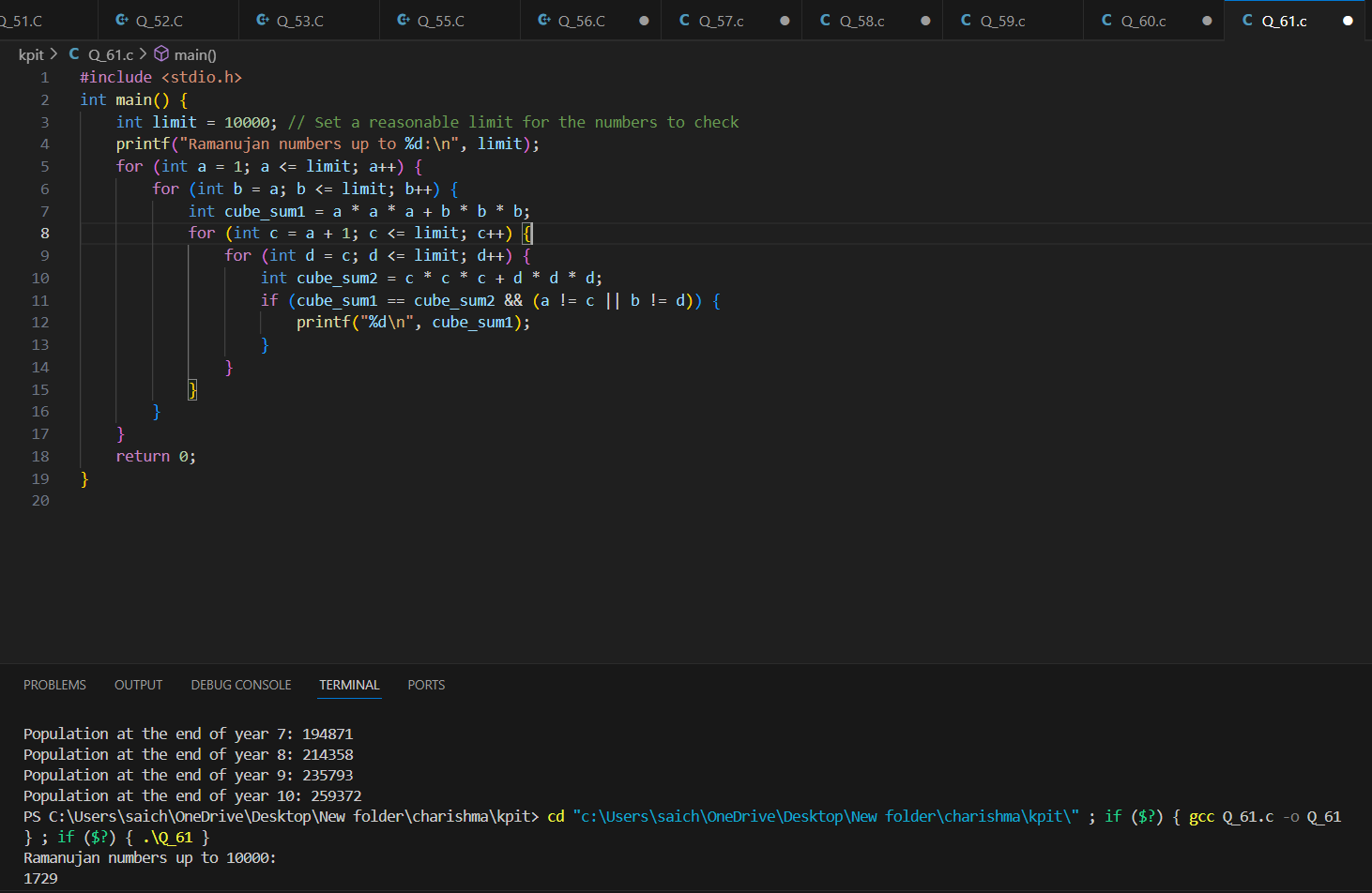
****

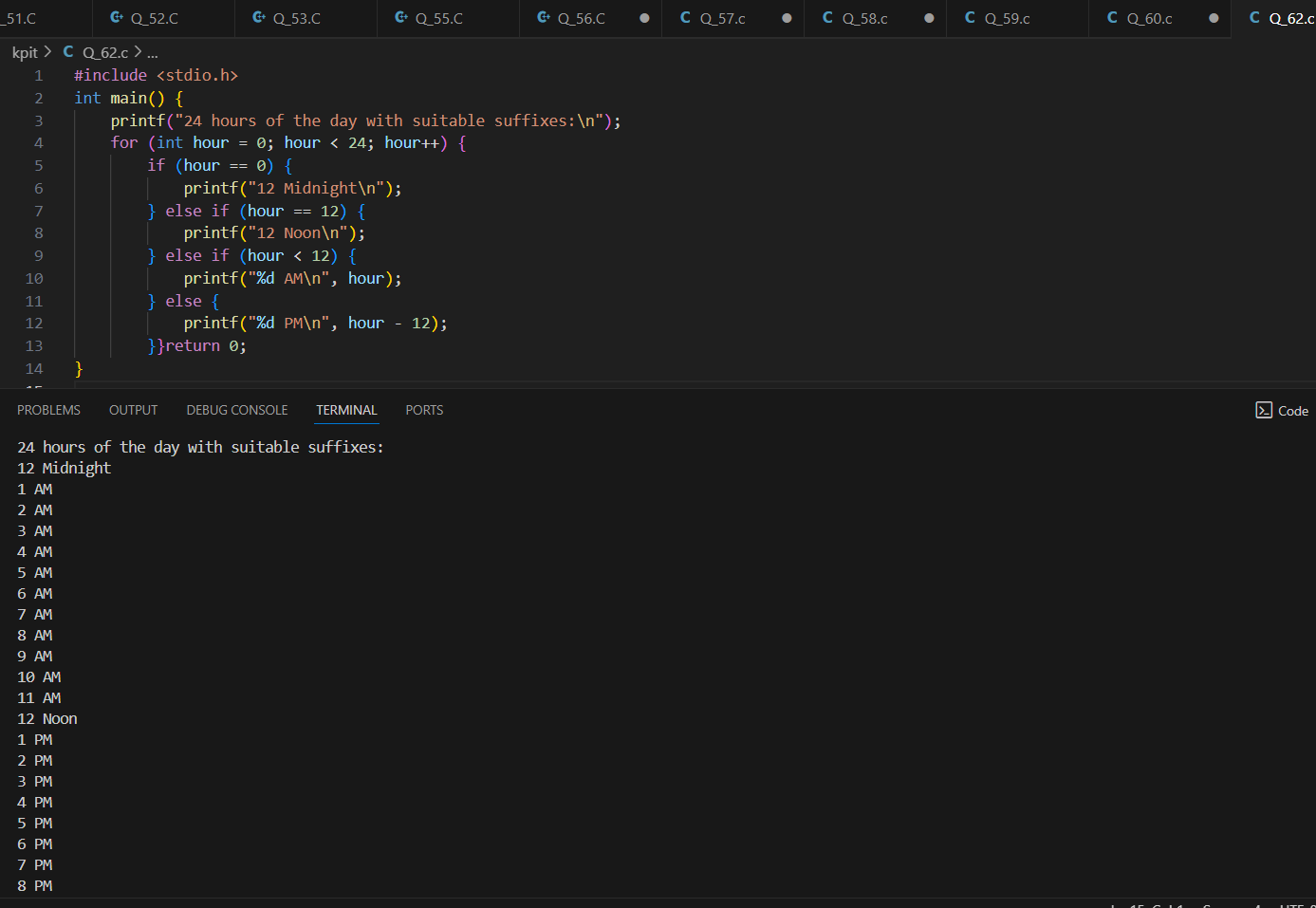
****

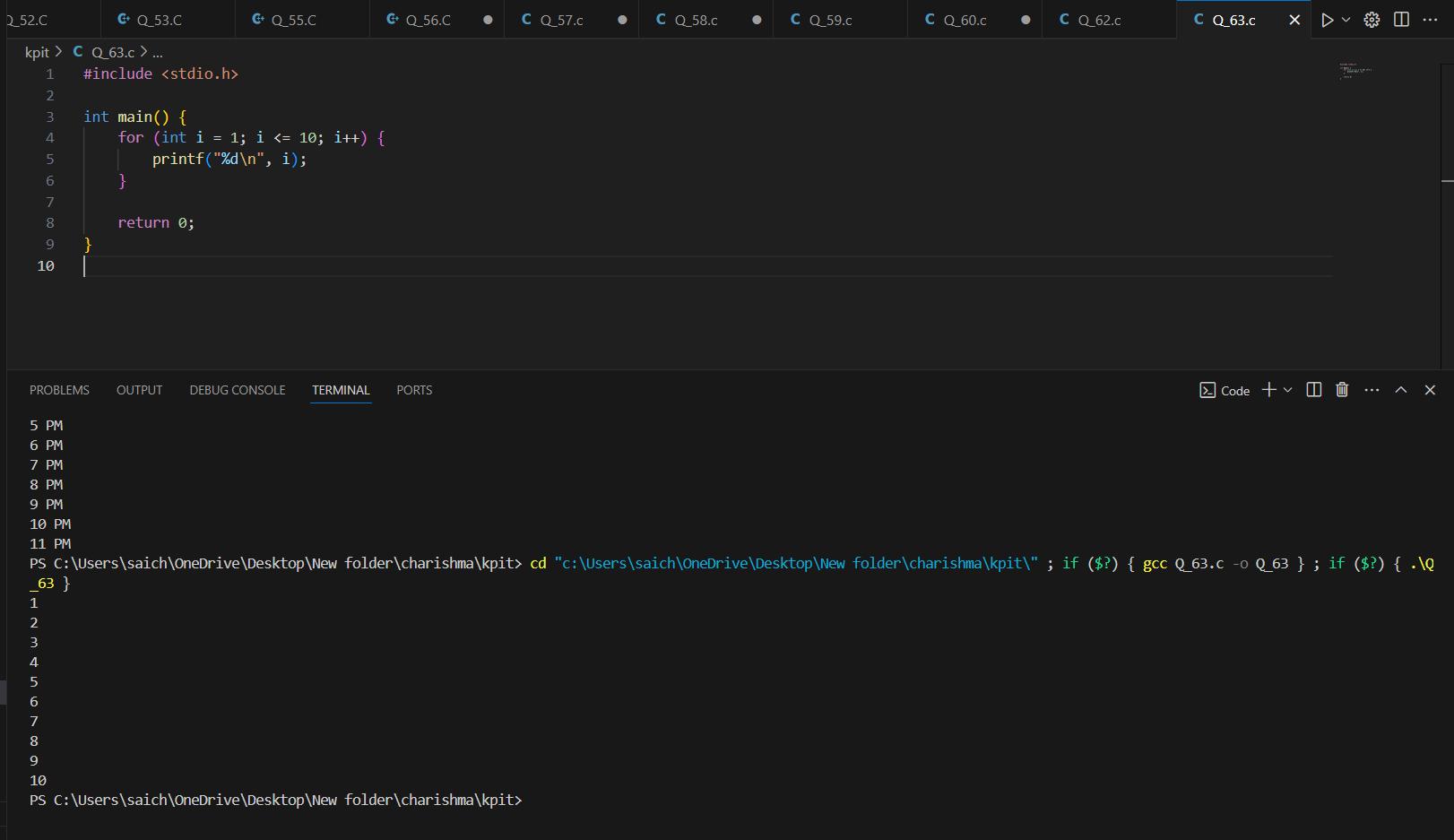
****

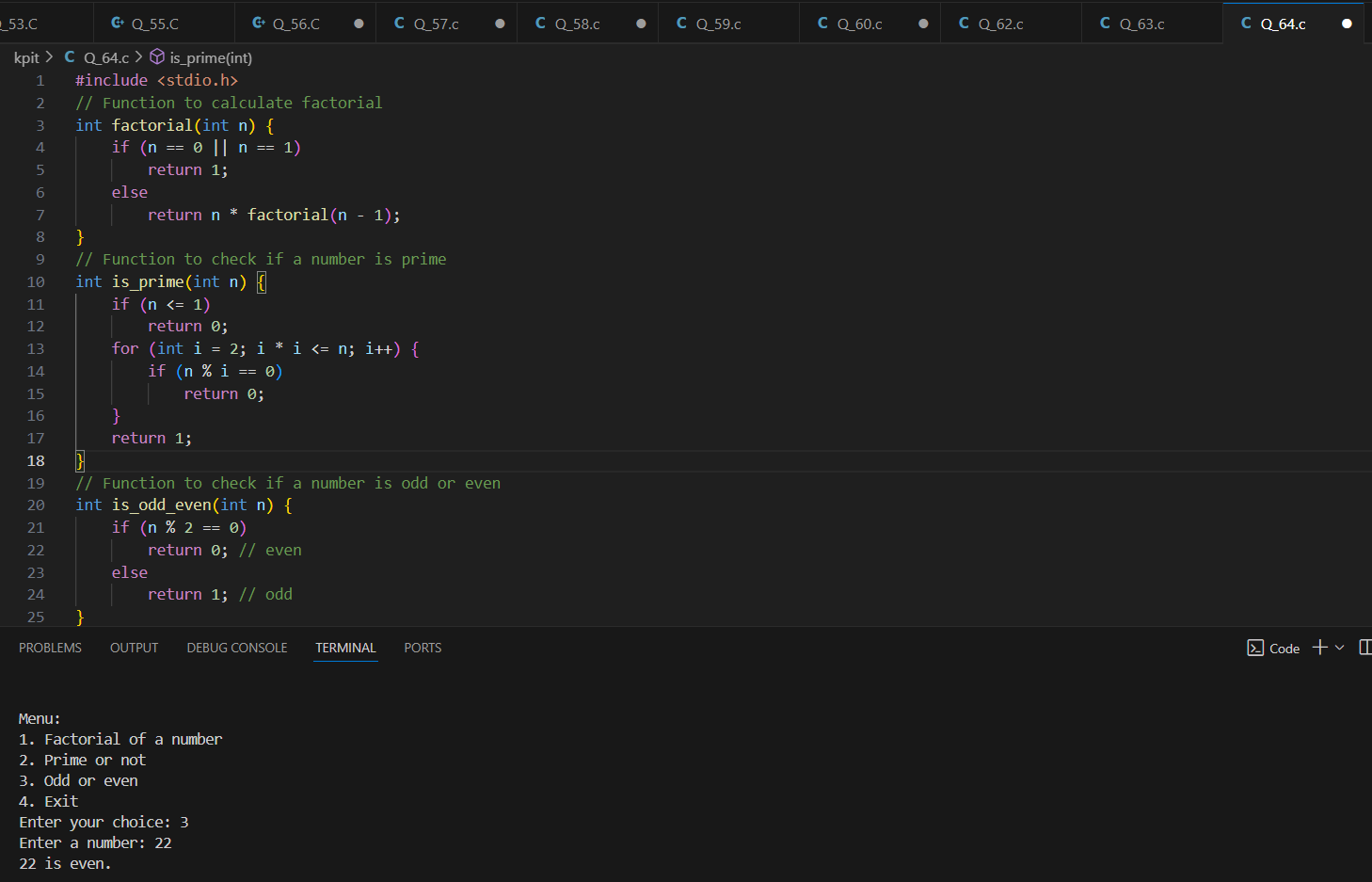
****

****

****

****

****

****

**(65)What will be the output of the following programs?**

**(a) # include <stdio.h>**

**int main( )**

**{**

**char suite = 3 ;**

**switch ( suite )**

**{**

**case 1 :**

**printf ( "Diamond\n" ) ;**

**case 2 :**

**printf ( "Spade\n" ) ;**

**default :**

**printf ( "Heart\n" ) ;**

**}**

**printf ( "I thought one wears a suite\n" ) ;**

**return 0 ;**

**}**

ANS:Heart

I thought one wears a suite

**(b) # include <stdio.h>**

**int main( )**

**{**

**int c = 3 ;**

**switch ( c )**

**{**

**case '3' :**

**printf ( "You never win the silver prize\n" ) ;**

**break ;**

**case 3 :**

**printf ( "You always lose the gold prize\n" ) ;**

**break ;**

**default :**

**printf ( "Of course provided you win a prize\n" ) ;**

**}**

**return 0 ;**

**}**

ANS:You always lose the gold prize

**(c) # include <stdio.h>**

**int main( )**

**{**

**int i = 3 ;**

**switch ( i )**

**{**

**case 0 :**

**printf ( "Customers are dicey\n" ) ;**

**case 1 + 2 :**

**printf ( "Markets are pricey\n" ) ;**

**case 4 / 2 :**

**printf ( "Investors are moody\n" ) ;**

**}**

**return 0 ;**

**}**

ANS: Markets are pricey

Investors are moody

**(d) # include <stdio.h>**

**int main( )**

**{**

**int k ;**

**float j = 2.0 ;**

**switch ( k = j + 1 )**

**{**

**case 3 :**

**printf ( "Trapped\n" ) ;**

**break ;**

**default :**

**printf ( "Caught!\n" ) ;**

**}**

**return 0 ;**

**}**

ANS: Caught!

**(e) # include <stdio.h>**

**int main( )**

**{**

**int ch = 'a' + 'b' ;**

**switch ( ch )**

**{**

**case 'a' :**

**case 'b' :**

**printf ( "Look at 10 ideas, 11th will occur to you\n" ) ;**

**case 'A' :**

**printf ( "If you have a good idea, project it\n" ) ;**

**case 'b' + 'a' :**

**printf ( "Have ideas, will fly\n" ) ;**

**}**

**return 0 ;**

**}**

ANS: Look at 10 ideas, 11th will occur to you

If you have a good idea, project it

Have ideas, will fly

**(66)Point out the errors, if any, in the following programs:**

**(a) # include <stdio.h>**

**int main( )**

**{**

**int suite = 1 ;**

**switch ( suite ) ;**

**{**

**case 0 ;**

**printf ( "Club\n" ) ;**

**case 1 ;**

**printf ( "Diamond\n" ) ;**

**}**

**return 0 ;**

**}**

ANS: The semicolon after the switch (suite) should not be there. It ends the switch statement prematurely. The case labels should not have semicolons; they should have colons.

**(b) # include <stdio.h>**

**int main( )**

**{**

**int temp ;**

**scanf ( "%d", &temp ) ;**

**switch ( temp )**

**{**

**case ( temp <= 20 ) :**

**printf ( "Ooooooohhhh! Damn cool!\n" ) ;**

**case ( temp > 20 && temp <= 30 ) :**

**printf ( "Rain rain here again!\n" ) ;**

**case ( temp > 30 && temp <= 40 ) :**

**printf ( "Wish I am on Everest\n" ) ;**

**default :**

**printf ( "Good old Nagpur weather\n" ) ;**

**}**

**return 0 ;**

**}**

ANS: Case labels in switch statements must be constant expressions. You cannot use relational expressions directly in case labels.

**(c) # include <stdio.h>**

**int main( )**

**{**

**float a = 3.5 ;**

**switch ( a )**

**{**

**case 0.5 :**

**printf ( "The art of C\n" ) ; break ;**

**case 1.5 :**

**printf ( "The spirit of C\n" ) ; break ;**

**case 2.5 :**

**printf ( "See through C\n" ) ; break ;**

**}**

**return 0 ;**

**}**

ANS: The switch statement cannot handle floating-point numbers directly. Case labels must be integer constants.

**(d) # include <stdio.h>**

**int main( )**

**{**

**int a = 3, b = 4, c ;**

**c = b – a ;**

**switch ( c )**

**{**

**case 1 || 2 :**

**printf ( "God give me a chance to change things\n" ) ;**

**break ;**

**case a || b :**

**printf ( "God give me a chance to run my show\n" ) ;**

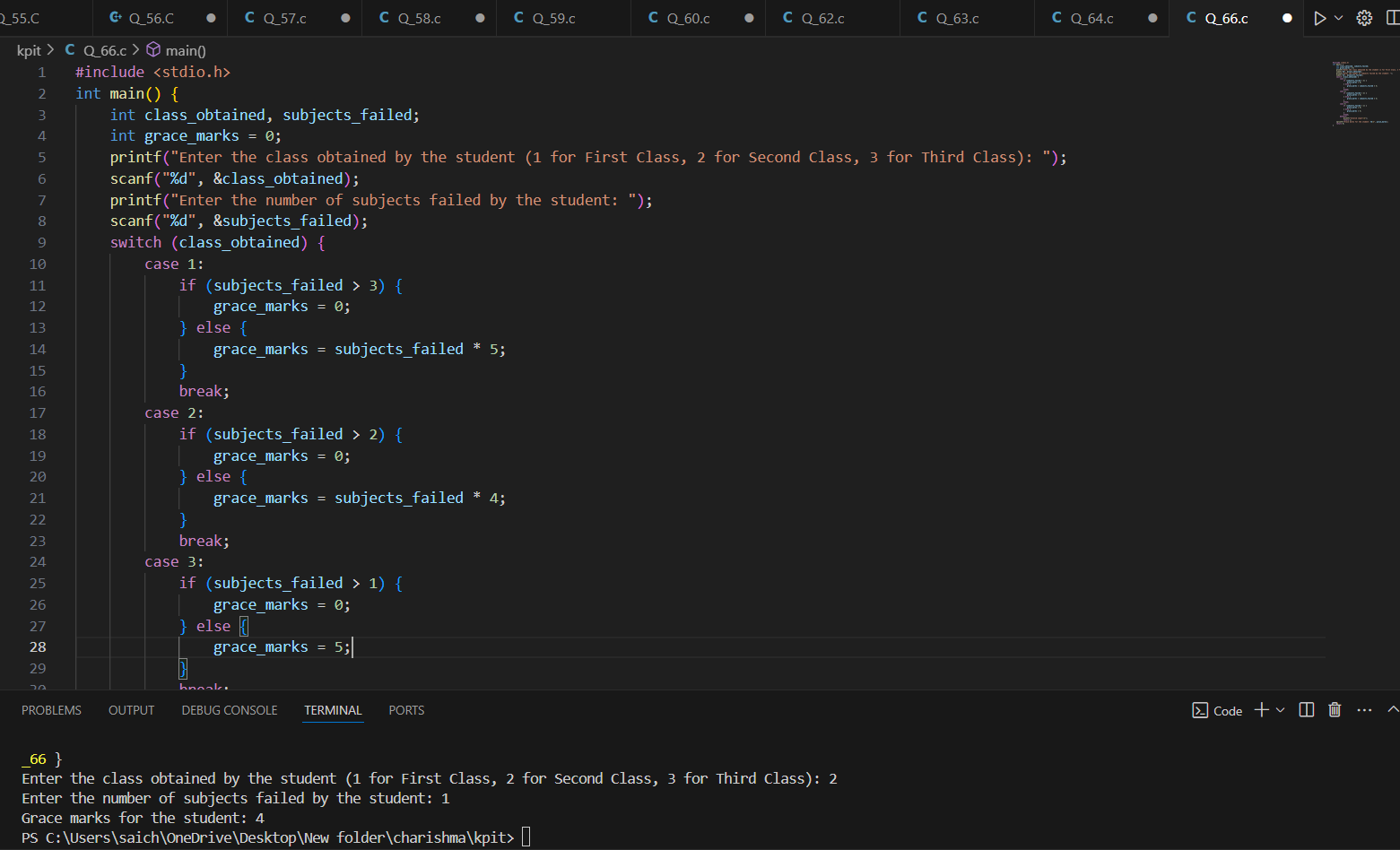
**break ;**

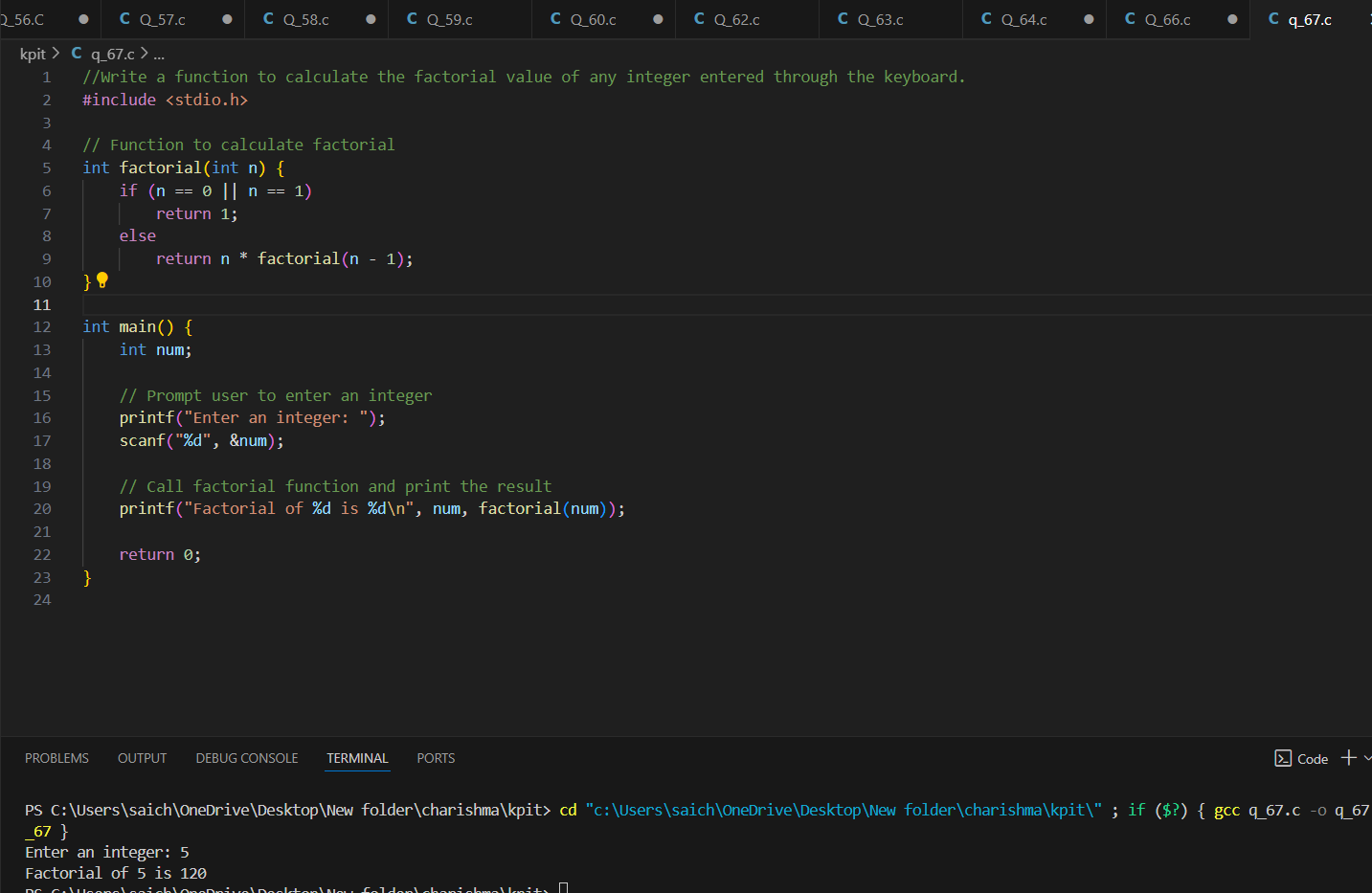
**}**

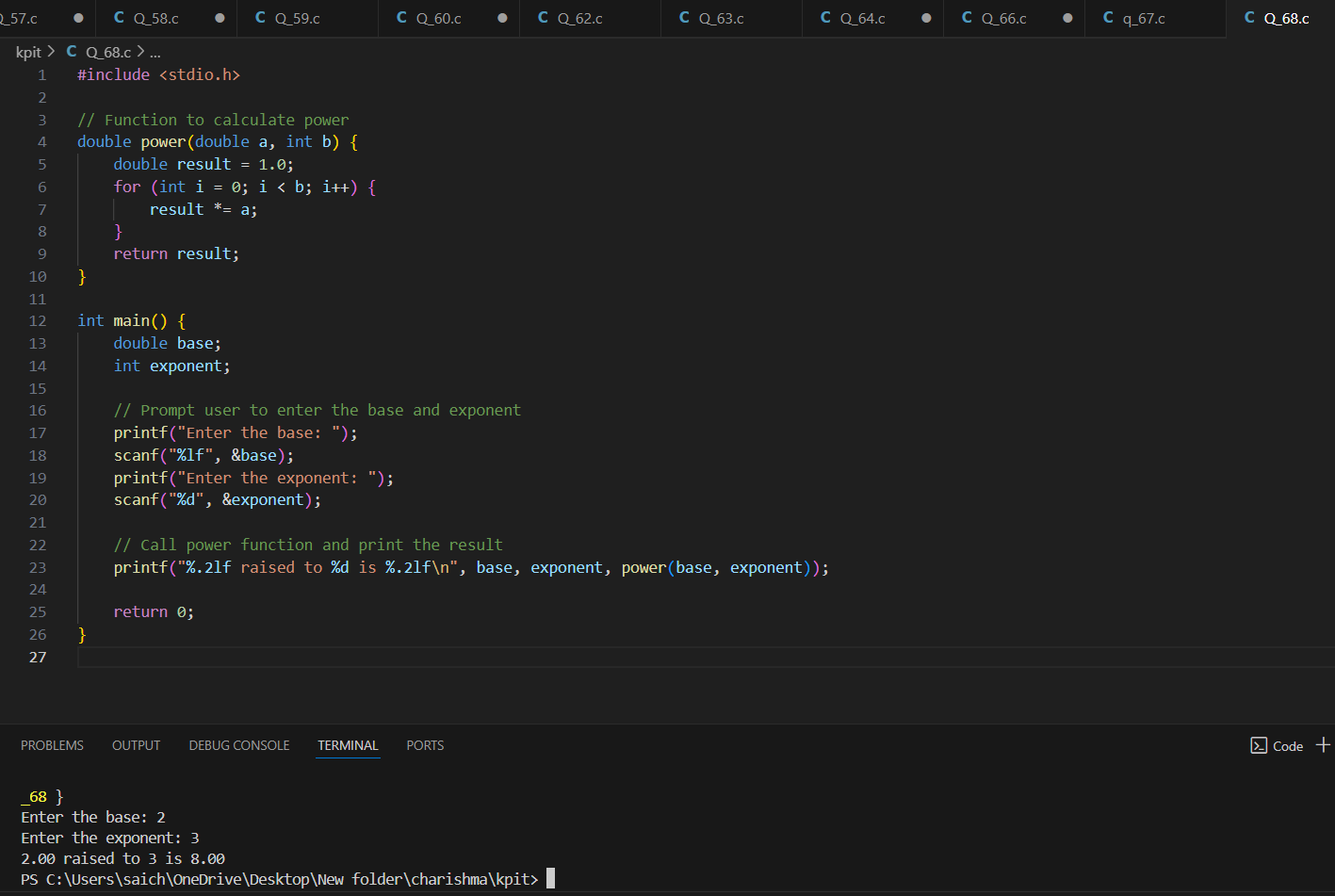
**return 0 ;**

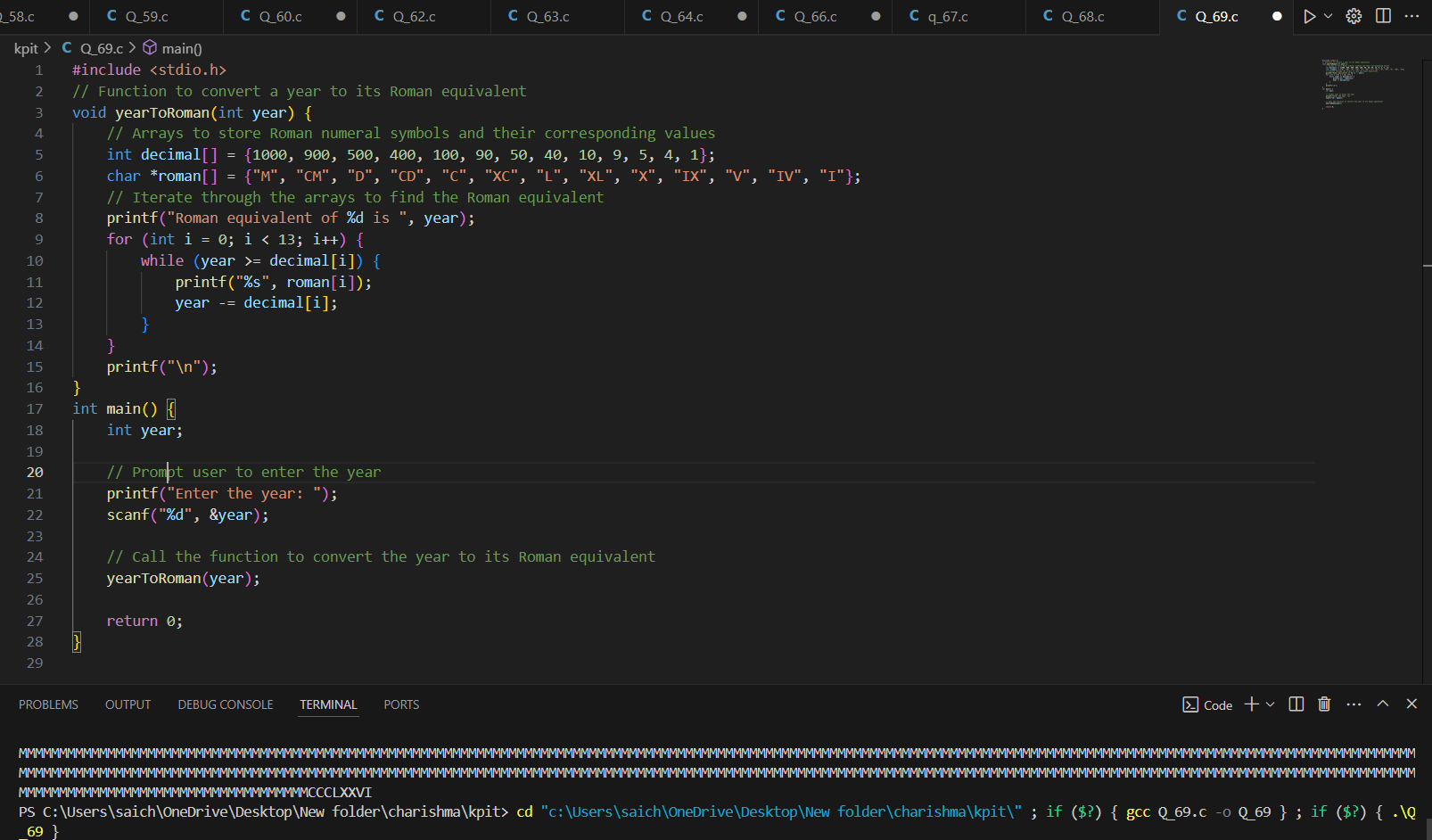
**}**

ANS: 1 || 2 is a logical OR operation which evaluates to 1. Thus, it is equivalent to case 1. a || b is also a logical OR operation that evaluates to 1 since a and b are non-zero. Thus, it is equivalent to case 1. The subtraction sign was incorrectly formatted as dash.









**(70) Point out the errors, if any, in the following programs:**

**(a) # include <stdio.h>**

**int addmult ( int, int )**

**int main( )**

**{**

**int i = 3, j = 4, k, l ;**

**k = addmult ( i, j ) ;**

**l = addmult ( i, j ) ;**

**printf ( "%d %d\n", k, l ) ;**

**return 0 ;**

**}**

**int addmult ( int x, int y )**

**{**

**int u, v ;**

**u = x + y ;**

**v = x \* y ;**

**return ( u, v ) ;**

**}**

**ANS:** The function addmult should have a prototype declaration. The return (u, v); statement returns the value of v only, which is not the intended behavior.

**(b) # include <stdio.h>**

**int main( )**

**{**

**int a ;**

**a = message( ) ;**

**return 0 ;**

**}**

**void message( )**

**{**

**printf ( "Learn from him online at ykanetkar.com\n" ) ;**

**return ;**

**}**

**ANS:** The message function is declared to return void, but it is used as if it returns an int.

**(c) # include <stdio.h>**

**int main( )**

**{**

**float a = 15.5 ;**

**char ch = 'C' ;**

**printit ( a, ch ) ;**

**return 0 ;**

**}**

**printit ( a, ch )**

**{**

**printf ( "%f %c\n", a, ch ) ;**

**}**

**ANS:** The printit function lacks a proper declaration and definition.

**(d) # include <stdio.h>**

**int main( )**

**{**

**let\_us\_c( )**

**{**

**printf ( "Learn C online…\n" ) ;**

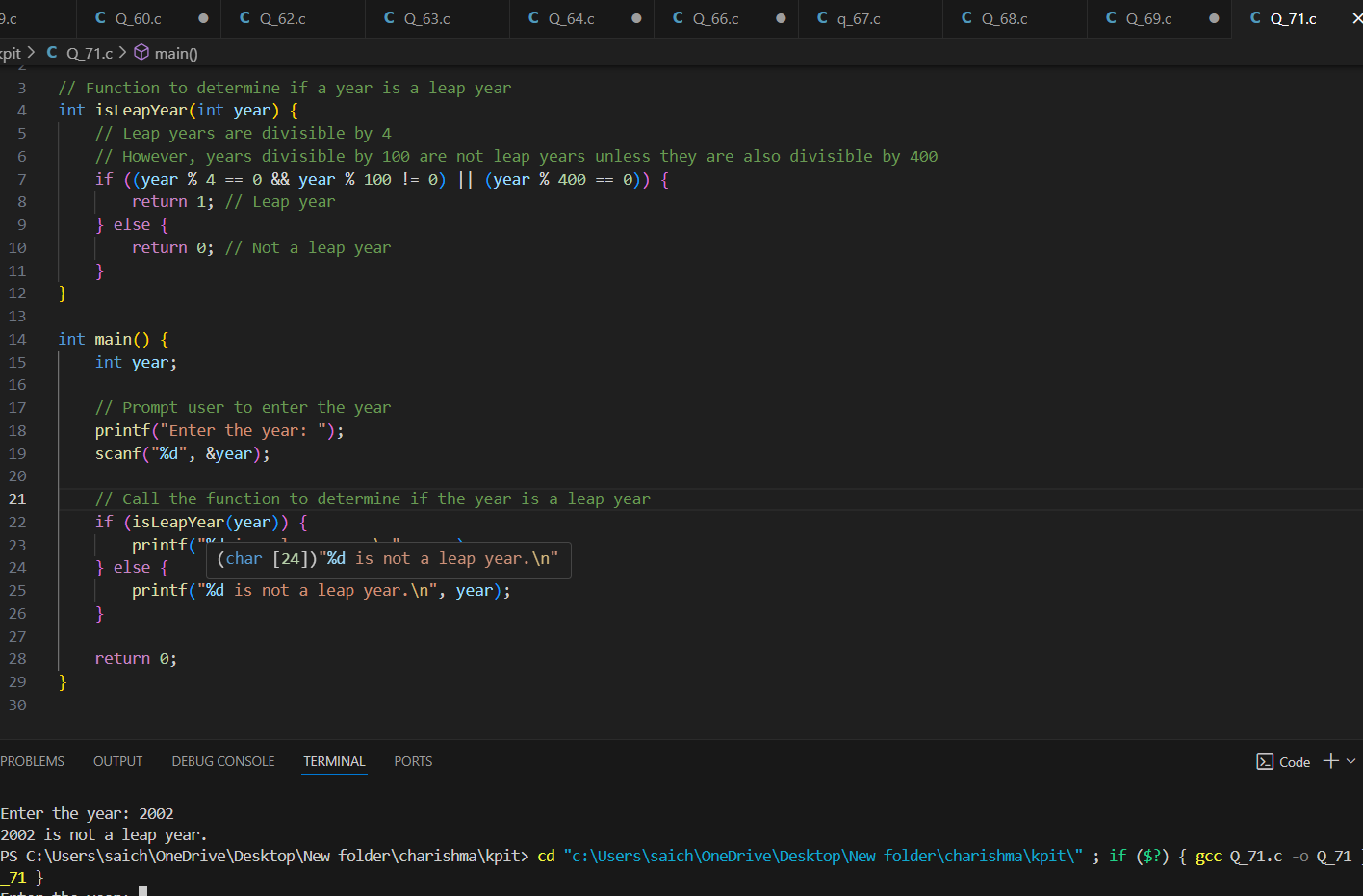
**printf ( "At ykanetkar.com\n" ) ;**

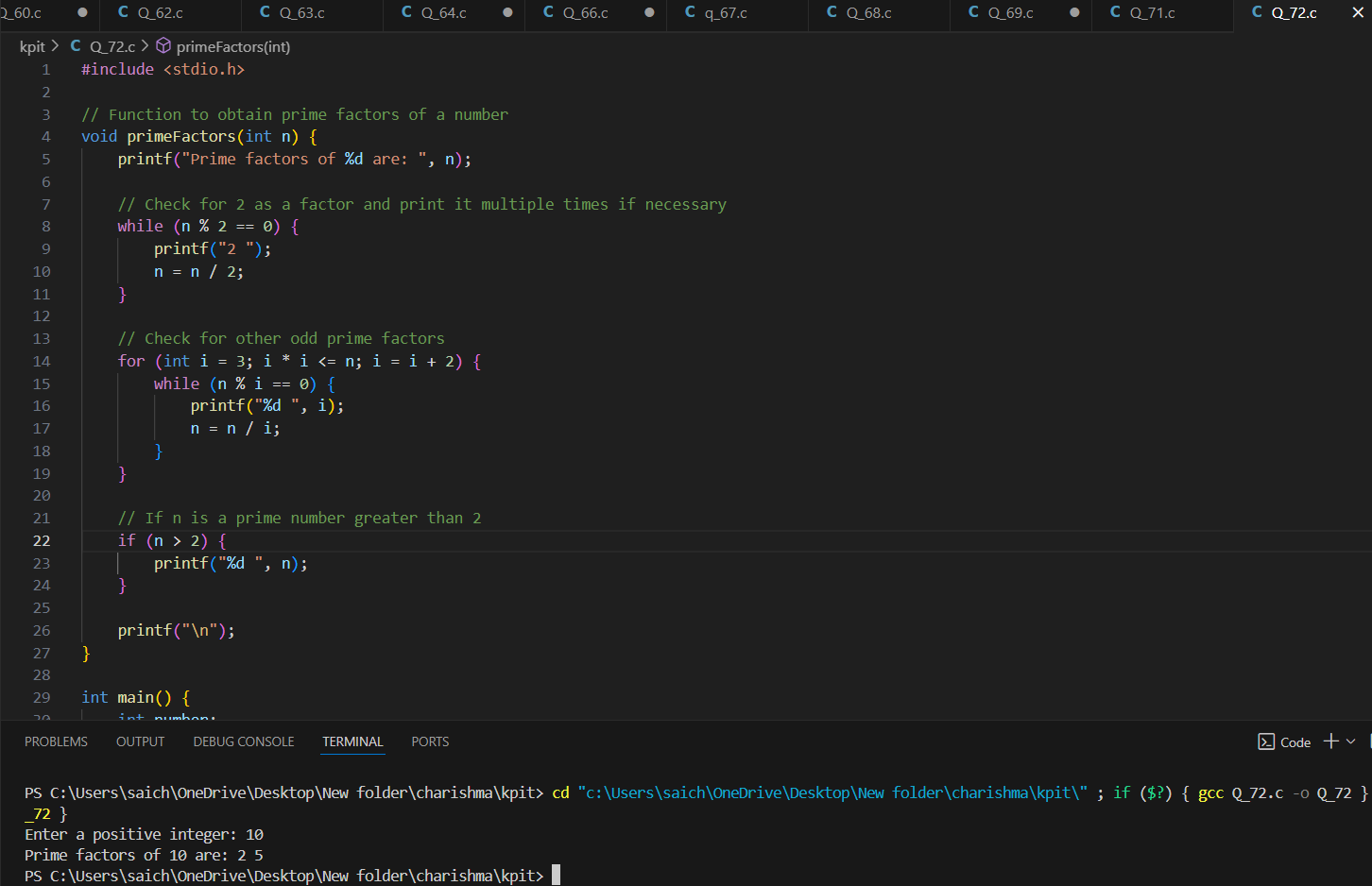
**}**

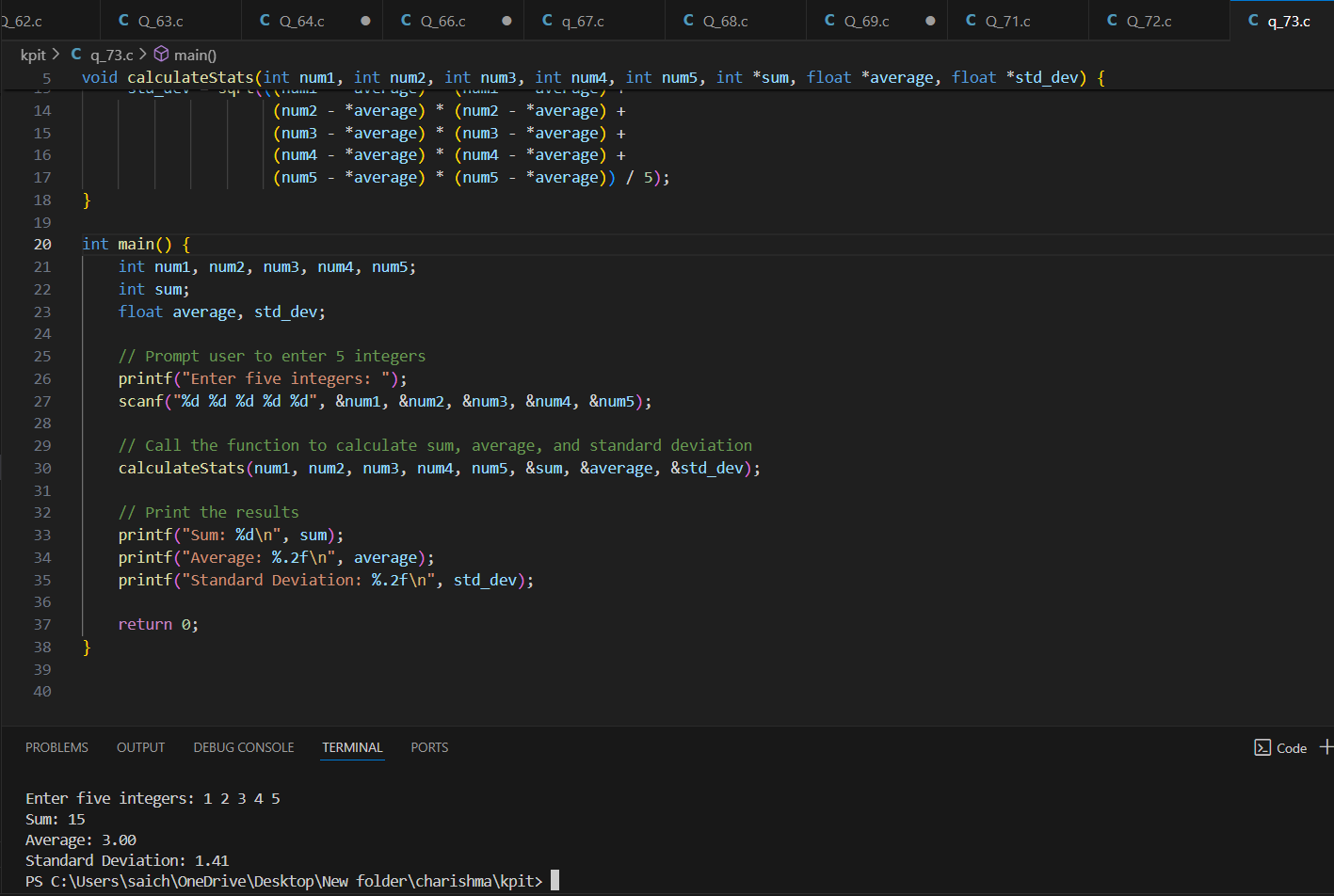
**return 0 ;**

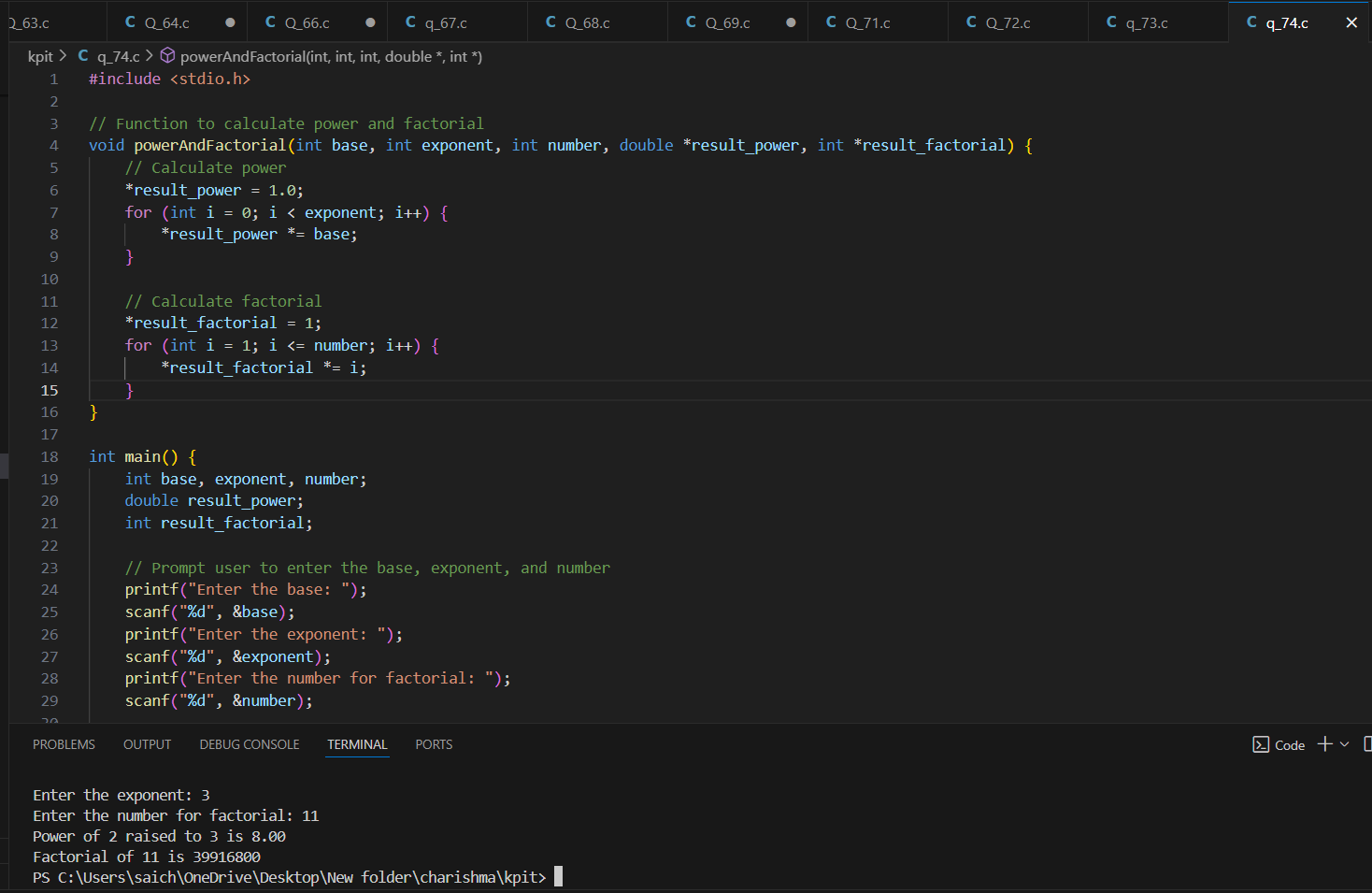
**}**

ANS: The let\_us\_c function is used without a proper declaration and definition.









**(75)What will be the output of the following programs?**

**(a) # include <stdio.h>**

**void fun ( int, int ) ;**

**int main( )**

**{**

**int i = 5, j = 2 ;**

**fun ( i, j ) ;**

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

**return 0 ;**

**}**

**void fun ( int i, int j )**

**{**

**i = i \* i ;**

**j = j \* j ;**

**}**

**ANS: 5 2**

**(b) # include <stdio.h>**

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

**int main( )**

**{**

**int i = 5, j = 2 ;**

**fun ( &i, &j ) ;**

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

**return 0 ;**

**}**

**void fun ( int \*i, int \*j )**

**{**

**\*i = \*i \* \*i ;**

**\*j = \*j \* \*j ;**

**}**

**ANS: 25 4**

**(c) # include <stdio.h>**

**int main( )**

**{**

**float a = 13.5 ;**

**float \*b, \*c ;**

**b = &a ; /\* suppose address of a is 1006 \*/**

**c = b ;**

**printf ( "%u %u %u\n", &a, b, c ) ;**

**printf ( "%f %f %f %f %f\n", a, \*(&a), \*&a, \*b, \*c ) ;**

**return 0 ;**

**}**

ANS: 1006 1006 1006

13.500000 13.500000 13.500000 13.500000 13.500000

(76) **Point out the errors, if any, in the following programs:**

**(a) # include <stdio.h>**

**int main( )**

**{**

**float p = 23.24 ;**

**int \*q, \*\*r ;**

**q = &p ;**

**r = &q ;**

**printf ( "%f %f\n", \*q, \*\*r ) ;**

**return 0 ;**

**}**

**ANS:** In this program, q is a pointer to an integer, but it's assigned the address of a float variable p. Similarly, r is a pointer to a pointer to an integer, but it's assigned the address of a pointer to a float variable q. This results in a type mismatch error.

**(b) # include <stdio.h>**

**int main( )**

**{**

**char ch = 'A' ;**

**int k = 35 ;**

**float a = 3.14 ;**

**printf ( "%p %p %p\n", &ch, &k, &a ) ;**

**return 0 ;**

**}**

**ANS: No error**

**(c) # include <stdio.h>**

**void function ( int \* ) ;**

**int main( )**

**{**

**int i = 35, \*z ;**

**z = function ( &i ) ;**

**printf ( "%d\n", z ) ;**

**return 0 ;**

**}**

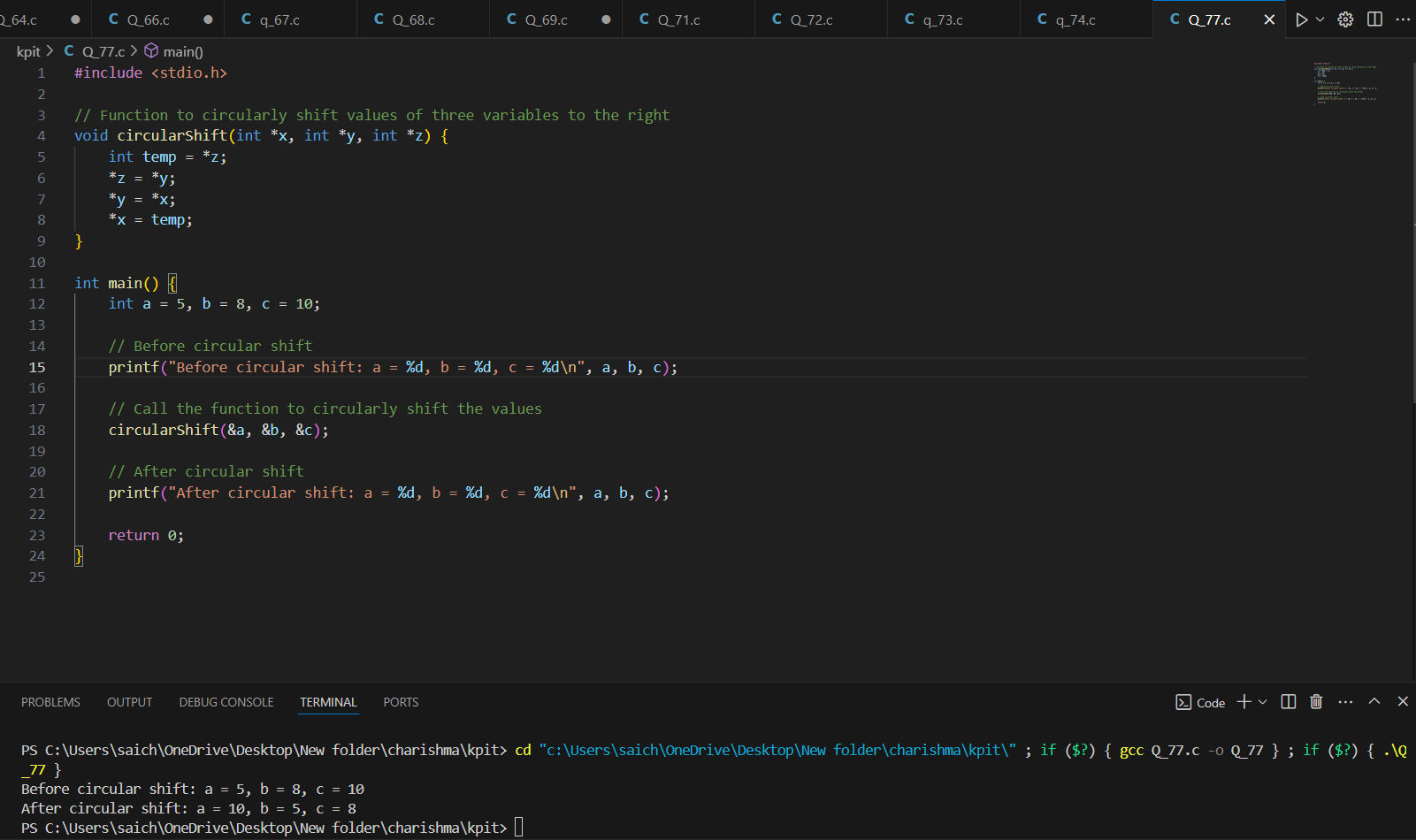
**void function ( int \*m )**

**{**

**return ( \*m + 2 ) ;**

**}**

**ANS:** Function Return Type Mismatch

****

**(78)**

#include <stdio.h>

// Function to convert weight from kilograms to grams, tons, and pounds

void convertWeight(double kilograms, double \*grams, double \*tons, double \*pounds) {

\*grams = kilograms \* 1000; // 1 kilogram = 1000 grams

\*tons = kilograms / 1000; // 1 kilogram = 0.001 tons

\*pounds = kilograms \* 2.20462; // 1 kilogram = 2.20462 pounds

}

int main() {

double kilograms, grams, tons, pounds;

// Prompt user to enter the weight in kilograms

printf("Enter the weight in kilograms: ");

scanf("%lf", &kilograms);

// Call the function to convert weight to grams, tons, and pounds

convertWeight(kilograms, &grams, &tons, &pounds);

// Print the results

printf("%.2f kilograms is equivalent to:\n", kilograms);

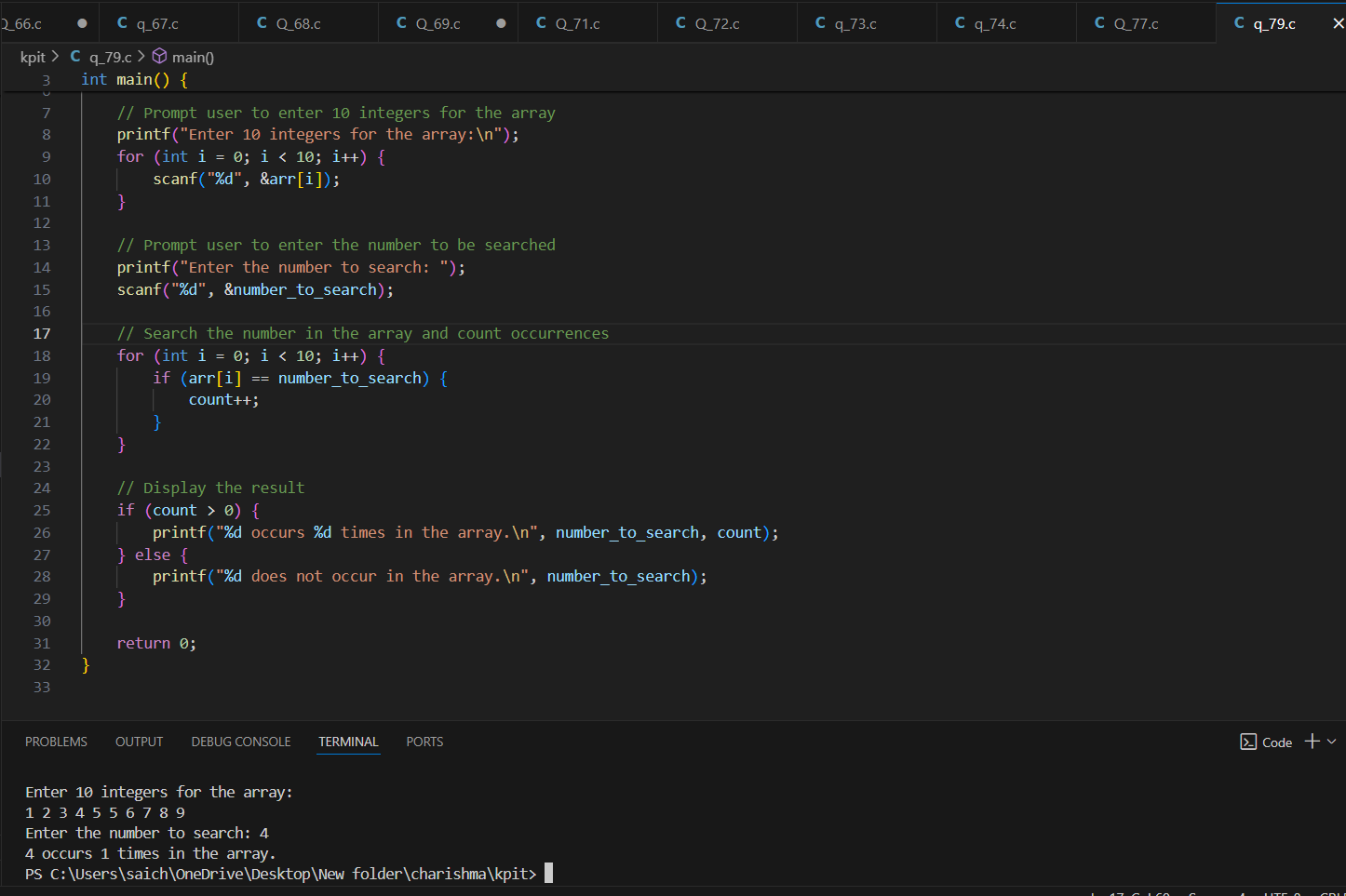
printf("%.2f grams\n", grams);

printf("%.2f tons\n", tons);

printf("%.2f pounds\n", pounds);

return 0;

}

**(79)** ****

**(80)Answer the following questions:**

**(a) Are the following array declarations correct?**

**int a (25) ;**

**int size = 10, b[ size ] ;**

Incorrect. Correct syntax: int a[25];

**(b) Which element of the array does this expression reference? num[ 4 ]**

It references the fifth element of the array num.

**(c) What is the difference between the 5’s in these two expressions?**

**int num[ 5 ] ;**

**num[ 5 ] = 11 ;**

The first expression declares an array of size 5, while the second tries to assign a value to the 6th element, which is out of bounds.

**(d) What will happen if you try to put so many values into an array when you initialize it that the size of the array is exceeded?**

It will result in a compilation error or warning.

**(e) What will happen if you put too few elements in an array when you initialize it?**

The remaining elements will be initialized to zero.

**(f) What will happen if you assign a value to an element of an array whose subscript exceeds the size of the array?**

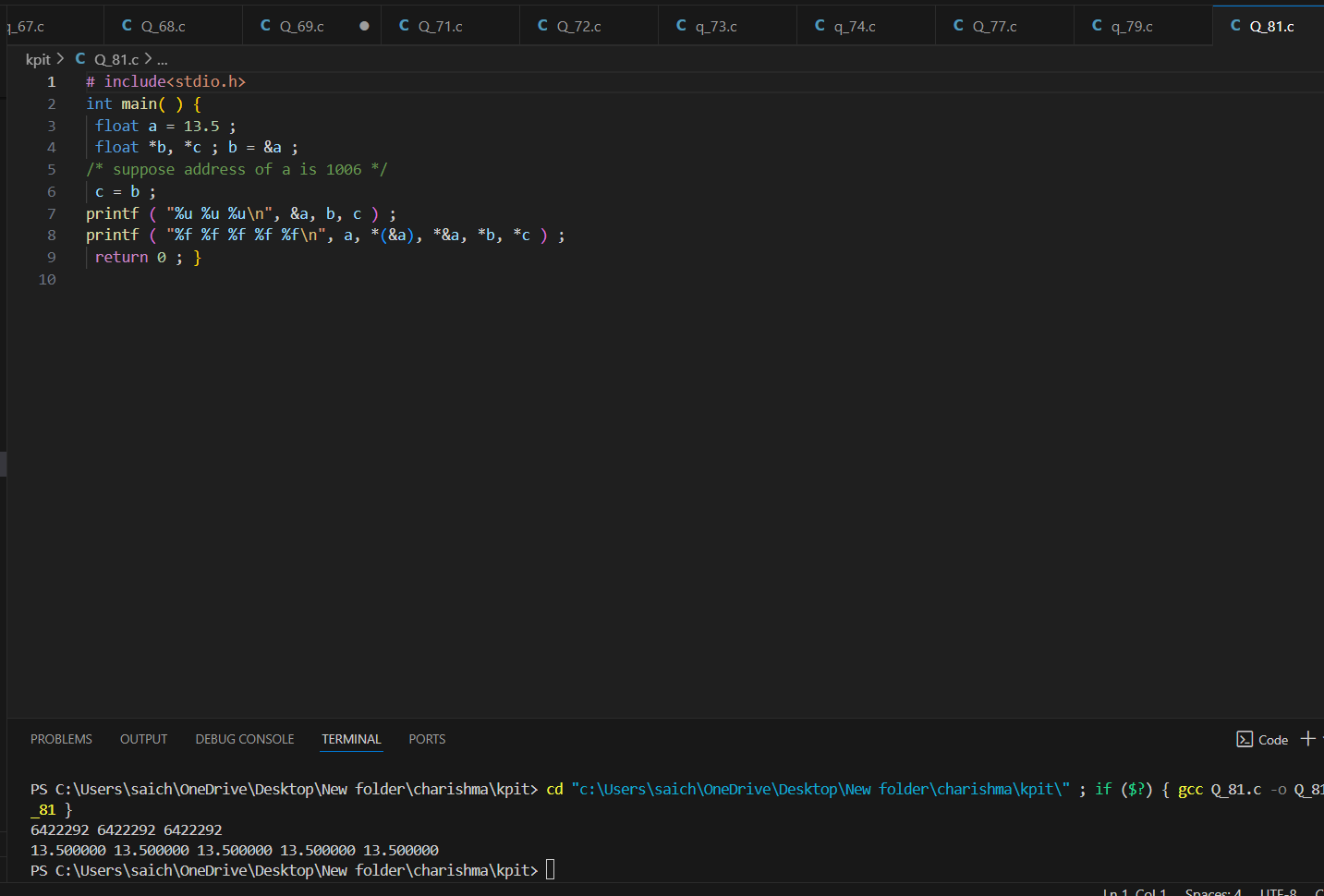
It will result in undefined behavior.

**(g) When you pass an array as an argument to a function, what actually gets passed?**

The address of the first element of the array.

**(h) If you don’t initialize a static array, what will its elements be set to?**

If declared at file scope, elements will be set to zero; if declared at block scope, elements will contain garbage values.



**(82) Point out the errors, if any, in the following programs:**

**(a) # include <stdio.h>**

**int main( )**

**{**

**int twod[ ][ ] = {**

**2, 4, 6, 8**

**} ;**

**printf ( "%d\n", twod ) ;**

**return 0 ;**

**}**

ANS: Incomplete array dimension: In C, when initializing a 2D array without specifying both dimensions, you must provide the size of the second dimension. In this case, the size of the second dimension should be specified as 4.

**(b) # include <stdio.h>**

**int main( )**

**{**

**int three[ 3 ][ ] = {**

**{ 2, 4, 3 }, { 6, 8, 2 }, { 2, 3, 1 }**

**} ;**

**printf ( "%d\n", three[ 1 ][ 1 ] ) ;**

**return 0 ;**

**}**

ANS: Incomplete array dimension: Similar to the previous error, when initializing a 2D array without specifying both dimensions, you must provide the size of the second dimension. In this case, the size of the second dimension should be specified based on the maximum number of elements in any row, which is 3.

**(83)Attempt the following questions:**

**(a) How will you initialize a three-dimensional array threed[ 3 ][ 2 ][ 3]?**

**How will you refer the first and last element in this array?**

**(b) Match the following with reference to the program segment given**

**below:**

**int i, j, = 25 ;**

**int \*pi, \*pj = & j ;**

**\*pj = j + 5 ;**

**j = \*pj + 5 ;**

**pj = pj ;**

**\*pi = i + j ;**

**Each integer quantity occupies 2 bytes of memory. The value assigned to i begins at (hexadecimal) address F9C and the value assigned to j begins at address F9E. Match the value represented by expression in left column with values in the right column.**

**1. &i a. 30**

**2. &j b. F9E**

**3. pj c. 35**

**4. \*pj d. FA2**

**5. i e. F9C**

**6. pi f. 67**

**7. \*pi g. unspecified**

**8. ( pi + 2 ) h. 65**

**9. (\*pi + 2) i. F9E**

**10. \* ( pi + 2 ) j. F9E**

**k. FAO**

**l. F9D**

**(c) Match the following with reference to the following program**

**segment:**

**int x[ 3 ][ 5 ] = {**

**{ 1, 2, 3, 4, 5 },**

**{ 6, 7, 8, 9, 10 },**

**{ 11, 12, 13, 14, 15 }**

**}, \*n = &x ;**

**1. \*( \*( x + 2 ) + 1) a. 9**

**2. \*( \*x + 2 ) + 5 b. 13**

**3. \*( \*( x + 1) ) c. 4**

**4. \*( \*( x ) + 2 ) + 1 d. 3**

**5. \* ( \*( x + 1 ) + 3 ) e. 2**

**6. \*n f. 12**

**7. \*( n +2 ) g. 14**

**8. (\*(n + 3 ) + 1 h. 7**

**9. \*(n + 5)+1 i. 1**

**10. ++\*n j. 8**

**k. 5**

**l. 10**

**m. 6**

**(d) Match the following with reference to the following program segment:**

**unsigned int arr[ 3 ][ 3 ] = {**

**{ 2, 4, 6 }, { 9, 1, 10 }, { 16, 64, 5 }**

**} ;**

**1. \*\*arr a. 64**

**2. \*\*arr < \*( \*arr + 2 ) b. 18**

**3. \*( arr + 2 ) / ( \*( \*arr + 1 ) > \*\*arr ) c. 6**

**4. \*( arr[ 1 ] + 1 ) | arr[ 1 ][ 2 ] d. 3**

**5. \*( arr[ 0 ] ) | \*( arr[ 2 ] ) e. 0**

**6. arr[ 1 ][ 1 ] < arr[ 0 ][ 1 ] f. 16**

**7. arr[ 2 ][ [ 1 ] & arr[ 2 ][ 0 ] g. 1**

**8. arr[ 2 ][ 2 ] | arr[ 0 ][ 1 ] h. 11**

**9. arr[ 0 ][ 1 ] ^ arr[ 0 ][ 2 ] i. 20**

**10. ++\*\*arr + --arr[ 1 ][ 1 ] j. 2**

**k. 5**

**l. 4**

**(e) Write a program to find if a square matrix is symmetric.**

**(f) Write a program to add two 6 x 6 matrices.**

**(g) Write a program to multiply any two 3 x 3 matrices.**

**(h) Given an array p[ 5 ], write a function to shift it circularly left by two positions. Thus, if the original array is { 15, 30, 28, 19, 61 } then after shifting it will be { 28, 19, 61, 15, 30 } Call this function for a 4 x 5 matrix and get its rows left shifted.  
ANS:**

(a) int threed[3][2][3] = {

{

{1, 2, 3},

{4, 5, 6}

},

{

{7, 8, 9},

{10, 11, 12}

},

{

{13, 14, 15},

{16, 17, 18}

}

};

(b)

1. &i - e. F9C
2. &j - l. F9E
3. pj - h. 65
4. \*pj - k. FA0
5. i - a. 30
6. pi - f. 67
7. \*pi - g. unspecified
8. (pi + 2) - j. F9E
9. (\*pi + 2) - c. 35
10. \*(pi + 2) - i. F9D

(c)

1. \*( \*(x + 2) + 1) - g. 14
2. \*( \*x + 2 ) + 5 - b. 18
3. \*( \*(x + 1) ) - e. 2
4. \*( \*(x) + 2 ) + 1 - d. 3
5. ( \*(x + 1) + 3 ) - a. 9
6. \*n - f. 12
7. \*(n +2 ) - j. 8
8. (\*(n + 3 ) + 1 - l. 10
9. \*(n + 5)+1 - i. 1
10. ++\*n - c. 4

(d)

1. \*\*arr - f. 16
2. \*\*arr < \*( \*arr + 2 ) - b. 18
3. \*( arr + 2 ) / ( \*( \*arr + 1 ) > \*\*arr ) - a. 64
4. \*( arr[ 1 ] + 1 ) | arr[ 1 ][ 2 ] - d. 3
5. \*( arr[ 0 ] ) | \*( arr[ 2 ] ) - e. 0
6. arr[ 1 ][ 1 ] < arr[ 0 ][ 1 ] - i. 20
7. arr[ 2 ][ [ 1 ] & arr[ 2 ][ 0 ] - h. 11
8. arr[ 2 ][ 2 ] | arr[ 0 ][ 1 ] - g. 1
9. arr[ 0 ][ 1 ] ^ arr[ 0 ][ 2 ] - k. 5
10. ++\*\*arr + --arr[ 1 ][ 1 ] - j. 2

(e)

#include <stdio.h>

#define SIZE 3

int isSymmetric(int mat[SIZE][SIZE]) {

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

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

if (mat[i][j] != mat[j][i]) {

return 0;

}

}

}

return 1;

}

int main() {

int mat[SIZE][SIZE] = {

{1, 2, 3},

{2, 4, 5},

{3, 5, 6}

};

if (isSymmetric(mat)) {

printf("The matrix is symmetric.\n");

} else {

printf("The matrix is not symmetric.\n");

}

return 0;

}

(f)

#include <stdio.h>

#define ROWS 6

#define COLS 6

void addMatrices(int mat1[][COLS], int mat2[][COLS], int result[][COLS]) {

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

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

result[i][j] = mat1[i][j] + mat2[i][j];

}

}

}

void printMatrix(int mat[][COLS]) {

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

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

printf("%d ", mat[i][j]);

}

printf("\n");

}

}

int main() {

int mat1[ROWS][COLS] = { /\* fill with values \*/ };

int mat2[ROWS][COLS] = { /\* fill with values \*/ };

int result[ROWS][COLS];

addMatrices(mat1, mat2, result);

printf("Resultant matrix:\n");

printMatrix(result);

return 0;

}

(g) #include <stdio.h>

#define SIZE 3

void multiplyMatrices(int mat1[][SIZE], int mat2[][SIZE], int result[][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] += mat1[i][k] \* mat2[k][j];

}

}

}

}

void printMatrix(int mat[][SIZE]) {

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

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

printf("%d ", mat[i][j]);

}

printf("\n");

}

}

int main() {

int mat1[SIZE][SIZE] = { /\* fill with values \*/ };

int mat2[SIZE][SIZE] = { /\* fill with values \*/ };

int result[SIZE][SIZE];

multiplyMatrices(mat1, mat2, result);

printf("Resultant matrix:\n");

printMatrix(result);

return 0;

}

(h)

#include <stdio.h>

Void circularShiftLeftByTwo(int arr[], int size) {

int temp1 = arr[0]; // Store the first element

int temp2 = arr[1]; // Store the second element

// Shift elements to the left

for (int i = 0; i < size - 2; i++) {

arr[i] = arr[i + 2];

}

// Place the stored elements at the end

arr[size - 2] = temp1;

arr[size - 1] = temp2;

}

int main() {

// Define a 4x5 matrix (2D array)

int matrix[4][5] = {

{15, 30, 28, 19, 61},

{10, 20, 30, 40, 50},

{55, 45, 35, 25, 15},

{5, 10, 15, 20, 25}

};

// Left shift each row of the matrix

printf("Matrix after left shifting each row:\n");

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

circularShiftLeftByTwo(matrix[i], 5); // Call the function for each row

// Print the shifted row

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

printf("%d ", matrix[i][j]);

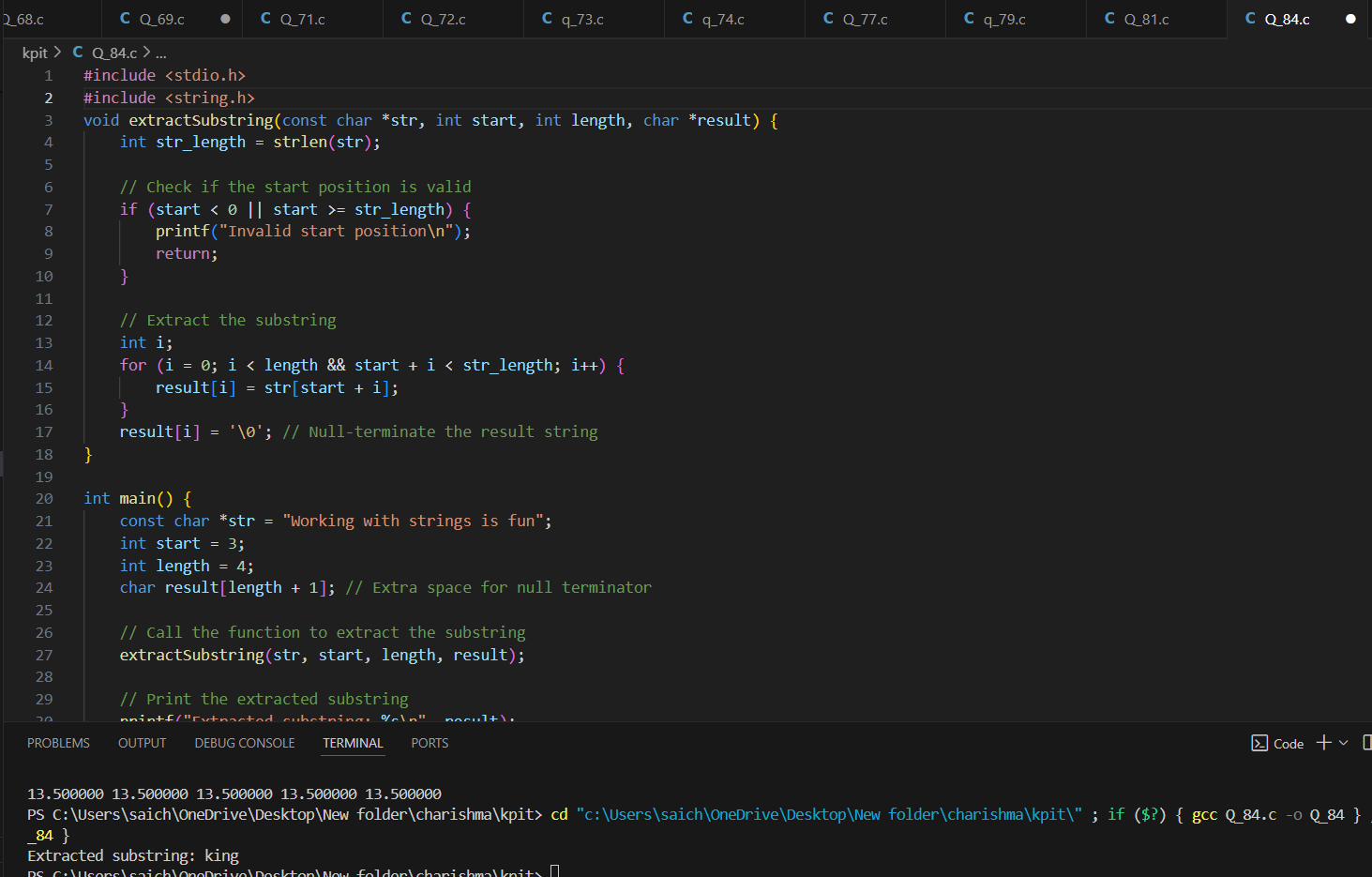
}

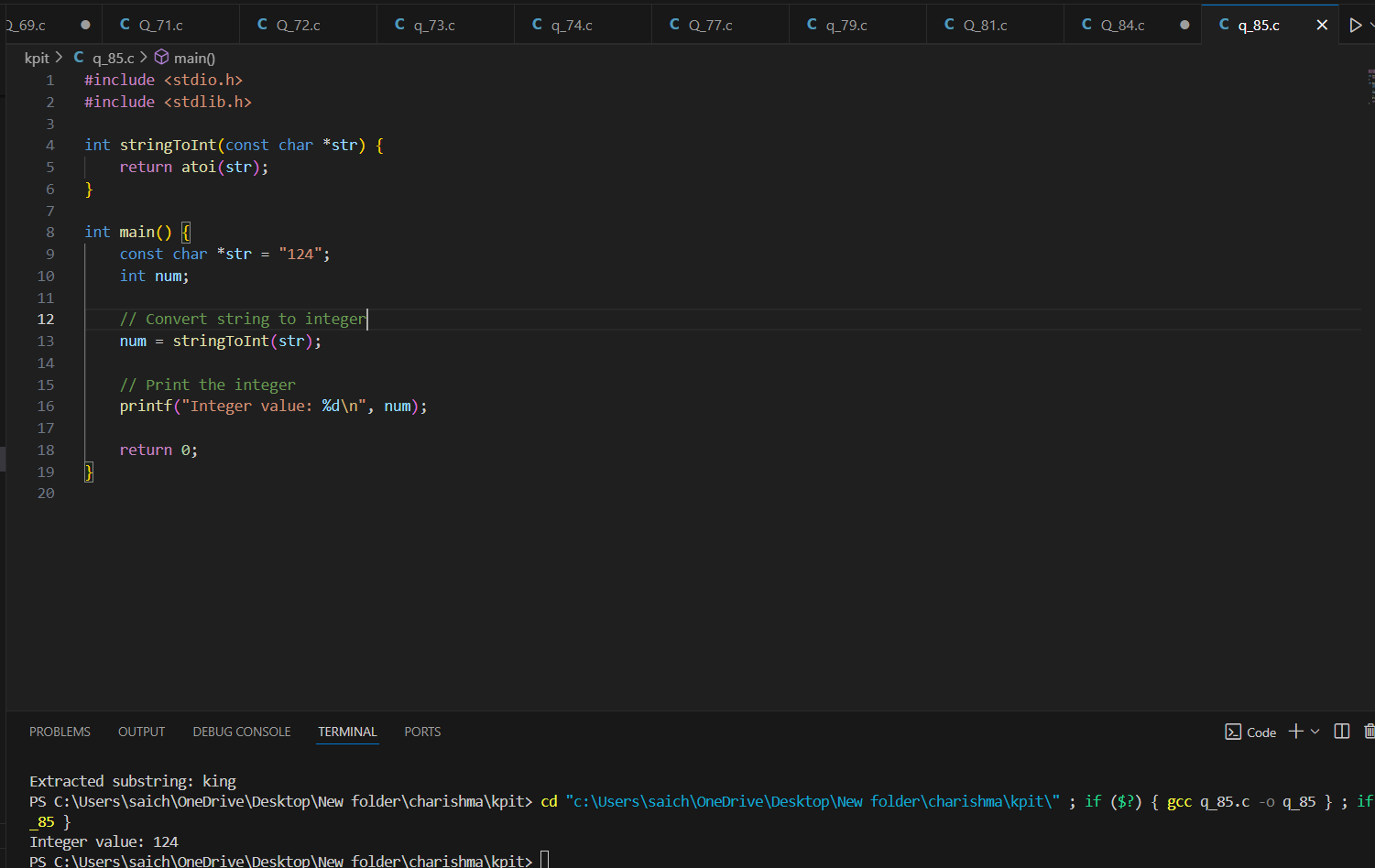
printf("\n");

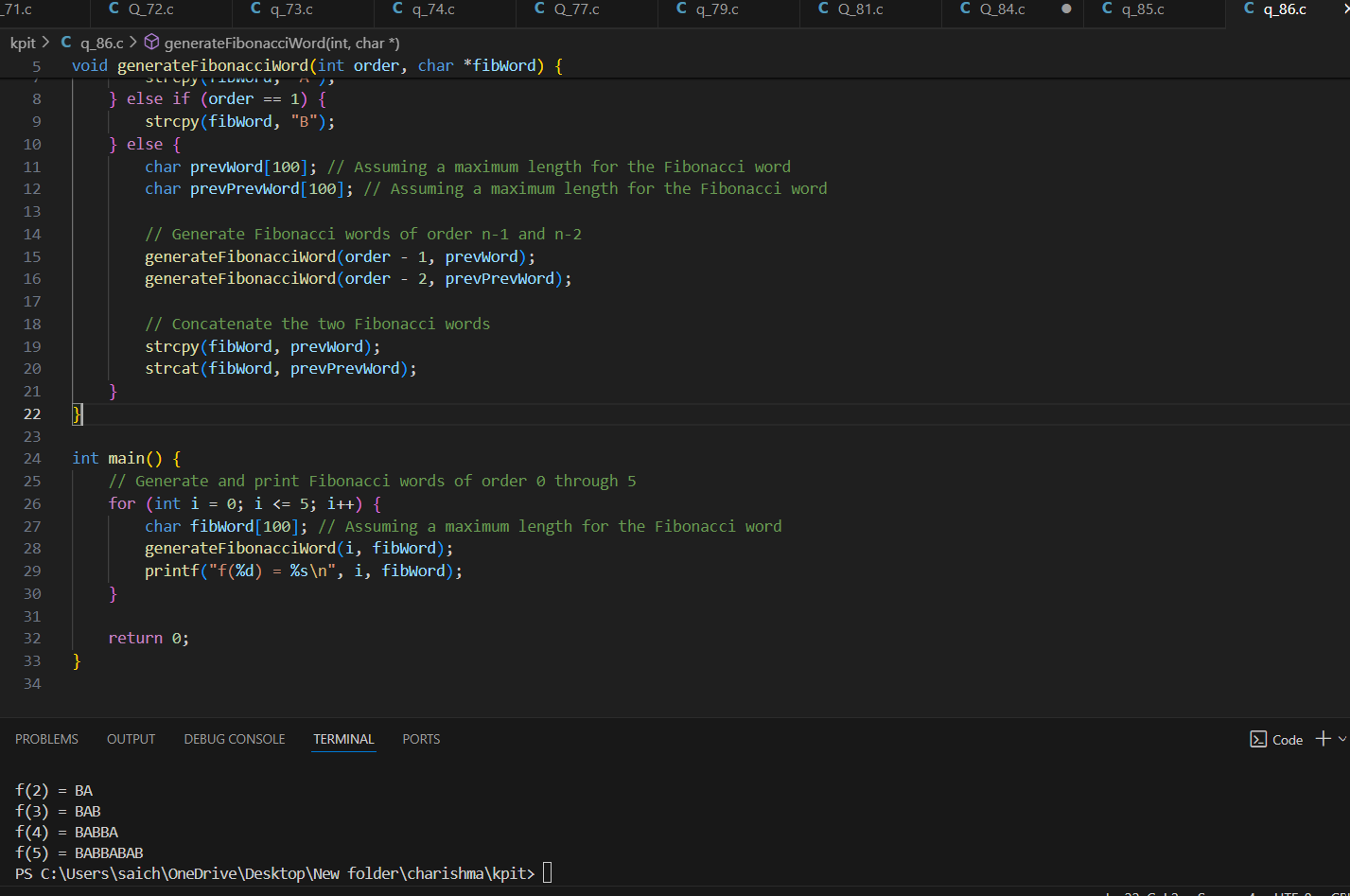
}

return 0;

}







**(87)What will be the output of the following programs?**

**(a) # include <stdio.h>**

**int main( )**

**{**

**char c[ 2 ] = "A" ;**

**printf ( "%c\n", c[ 0 ] ) ;**

**printf ( "%s\n", c ) ;**

**return 0 ;**

**}**

**ANS: A**

**(b) # include <stdio.h>**

**int main( )**

**{**

**char s[ ] = "Get organized! Learn C!!" ;**

**printf ( "%s\n", &s[ 2 ] ) ;**

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

**printf ( "%s\n", &s ) ;**

**printf ( "%c\n", s[ 2 ] ) ;**

**return 0 ;**

**}**

**ANS:** Get organized! Learn C!!

Get organized! Learn C!!

Get organized! Learn C!!

t

**(c) # include <stdio.h>**

**int main( )**

**{**

**char s[ ] = "Borrowers of books spoil the symmetry of shelves" ;**

**int i = 0 ;**

**while ( s[ i ] != 0 )**

**{**

**printf ( "%c %c\n", s[ i ], \*( s + i ) ) ;**

**printf ( "%c %c\n", i[ s ], \*( i + s ) ) ;**

**i++ ;**

**}**

**return 0 ;**

**}**

**ANS:** **B B**

**B B**

**o o**

**o o**

**r r**

**r r**

**r r**

**r r**

**o o**

**o o**

**w w**

**w w**

**e e**

**e e**

**r r**

**r r**

**s s**

**s s**

**(space)**

**(space)**

**o o**

**o o**

**f f**

**f f**

**(space)**

**(space)**

**b b**

**b b**

**o o**

**o o**

**o o**

**o o**

**k k**

**k k**

**s s**

**s s**

**(space)**

**(space)**

**s s**

**s s**

**p p**

**p p**

**o o**

**o o**

**i i**

**i i**

**l l**

**l l**

**(space)**

**(space)**

**t t**

**t t**

**h h**

**h h**

**e e**

**e e**

**(space)**

**(space)**

**s s**

**s s**

**y y**

**y y**

**m m**

**m m**

**m m**

**m m**

**e e**

**e e**

**t t**

**t t**

**r r**

**r r**

**y y**

**y y**

**(space)**

**(space)**

**o o**

**o o**

**f f**

**f f**

**(space)**

**(space)**

**s s**

**s s**

**h h**

**h h**

**e e**

**e e**

**l l**

**l l**

**v v**

**v v**

**e e**

**e e**

**s s**

**s s**

**(d) # include <stdio.h>**

**int main( )**

**{**

**char str1[ ] = { ’H’, ’e’, ’l’, ’l’, ’o’, 0 } ;**

**char str2[ ] = "Hello" ;**

**printf ( "%s\n", str1 ) ;**

**printf ( "%s\n", str2 ) ;**

**return 0 ;**

**}**

**ANS:** Hello

Hello

**(e) # include <stdio.h>**

**int main( )**

**{**

**printf ( 5 + "Good Morning " ) ;**

**printf ( "%c\n", "abcdefgh"[ 4 ] ) ;**

**return 0 ;**

**}**

**ANS:** Morning

e

**(f) # include <stdio.h>**

**int main( )**

**{**

**printf ( "%d %d %d\n", sizeof ( ’3’ ), sizeof ( "3" ), sizeof ( 3 ) ) ;**

**return 0 ;**

**}**

ANS: 1 2 4

**(88) Fill in the blanks:**

**(a) "A" is a \_\_\_\_\_\_\_\_\_\_\_ whereas ’A’ is a \_\_\_\_\_\_\_\_\_\_\_\_.**

**(b) A string is terminated by a \_\_\_\_\_\_ character.**

**(c) The array char name[ 10 ] can consist of a maximum of \_\_\_\_\_\_ characters.**

**(d) The array elements are always stored in \_\_\_\_\_\_\_\_\_ memory locations.**

**ANS:**

(a) "A" is a string literal whereas 'A' is a character literal.

(b) A string is terminated by a null character ('\0') character.

(c) The array char name[10] can consist of a maximum of 9 characters (plus 1 for the null terminator).

(d) The array elements are always stored in contiguous memory locations.

**(89)Answer the following questions:**

**(a) A preprocessor directive is:**

**1. A message from compiler to the programmer**

**2. A message from compiler to the linker**

**3. A message from programmer to the preprocessor**

**4. A message from programmer to the microprocessor**

**(b) Which of the following are correctly formed #define statements?**

**#define INCH PER FEET 12**

**#define SQR (X) ( X \* X )**

**#define SQR(X) X \* X**

**#define SQR(X) ( X \* X )**

**(c) State True or False:**

**1. A macro must always be written in capital letters.**

**2. A macro must always be accommodated in a single line.**

**3. After preprocessing when the program is sent for compilation**

**the macros are removed from the expanded source code.**

**4. Macros with arguments are not allowed.**

**5. In a macro call the control is passed to the macro.**

**(d) A header file is:**

**1. A file that contains standard library functions**

**2. A file that contains function declarations and macros**

**3. A file that contains user-defined functions**

**4. A file that is present in current working directory**

**(e) All macro substitutions in a program are done:**

**1. Before compilation of the program**

**2. After compilation of the program**

**3. During execution of the program**

**4. During linking of the program  
ANS:**

(a) A message from compiler to the programmer

(b) #define SQR(X) ( X \* X )

(c)

1. False
2. True
3. True
4. False
5. True

(d) A file that contains function declarations and macros

(e) Before compilation of the program

**(90)What will be the output of the following programs?**

**(a) # include <stdio.h>**

**int main( )**

**{**

**int i = 2 ;**

**# ifdef DEF**

**i \*= i ;**

**# else**

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

**# endif**

**return 0 ;**

**}**

**ANS:2**

**(b) # include <stdio.h>**

**# define PRODUCT(x) ( x \* x )**

**int main( )**

**{**

**int i = 3, j, k, l ;**

**j = PRODUCT( i + 1 ) ;**

**k = PRODUCT( i++ ) ;**

**l = PRODUCT ( ++i ) ;**

**printf ( "%d %d %d %d\n", i, j, k, l ) ;**

**return 0 ;**

**}**

**ANS:** 6 16 9 36

**(c) # include <stdio.h>**

**# define PI 3.14**

**# define AREA( x, y, z ) ( PI \* x \* x + y \* z ) ;**

**int main ( )**

**{**

**float a = AREA ( 1, 5, 8 ) ;**

**float b = AREA ( AREA ( 1, 5, 8 ), 4, 5 ) ;**

**printf ( " a = %f\n", a ) ;**

**printf ( " b = %f\n", b ) ;**

**return 0 ;**

**}**

**ANS:**

a = 43.140000

b = 5983.168457

**(91)** **State whether the following statements are True or False:**

**(a) The value of an automatic storage class variable persists between various function invocations.**

**(b) If the CPU registers are not available, the register storage class variables are treated as static storage class variables.**

**(c) If we try to use register storage class for a float variable the compiler will report an error message.**

**(d) The default value for automatic variable is zero.**

**(e) The life of static variable is till the control remains within the block in which it is defined.**

**(f) If a global variable is to be defined, then the extern keyword is necessary in its declaration.**

**(g) The address of register variable is not accessible.**

**ANS:** (a) False  
(b) False  
(c) True  
(d) False  
(e) False  
(f) False  
(g) True

**(92)What will be the output of the following programs?**

**(a) # include <stdio.h>**

**int i = 0 ;**

**void val( ) ;**

**int main( )**

**{**

**printf ( "main's i = %d\n", i ) ;**

**i++ ;**

**val( ) ;**

**printf ( "main's i = %d\n", i ) ;**

**val( ) ;**

**return 0 ;**

**}**

**void val( )**

**{**

**i = 100 ;**

**printf ( "val's i = %d\n", i ) ;**

**i++ ;**

**}**

**(b) # include <stdio.h>**

**int main( )**

**{**

**static int count = 5 ;**

**printf ( "count = %d\n", count-- ) ;**

**if ( count != 0 )**

**main( ) ;**

**return 0 ;**

**}**

**(c) # include <stdio.h>**

**void fnc( ) ;**

**int main( )**

**{**

**func( ) ;**

**func( ) ;**

**return 0 ;**

**}**

**void func( )**

**{**

**auto int i = 0 ;**

**register int j = 0 ;**

**static int k = 0 ;**

**i++ ; j++ ; k++ ;**

**printf ( "%d % d %d\n", i, j, k ) ;**

**}**

**(d) # include <stdio.h>**

**int x = 10 ;**

**int main( )**

**{**

**int x = 20 ;**

**{**

**int x = 30 ;**

**printf ( "%d\n", x ) ;**

**}**

**printf ( "%d\n", x ) ;**

**return 0 ;**

**}**

**ANS:**

(a) main's i = 0

val's i = 100

main's i = 101

val's i = 100

(b) count = 5

count = 4

count = 3

count = 2

count = 1

count = 0

(c) 1 1 1

1 1 1

(d) 30

20

**(93)Write macro definitions for the following: 1. To test whether a character is a lowercase letter or not. 2. To test whether a character is an uppercase letter or not. 3. To test whether a character is an alphabet or not. Make use of the macros you defined in 1 and 2 above. 4. To obtain the bigger of two numbers.**

**Ans:**

# include

#define ISUPPER(x) ( x >= 65 && x <= 90 ? 1 : 0 )

#define ISLOWER(x) ( x >= 97 && x <= 122 ? 1 : 0 )

#define ISALPHA(x) ( ISUPPER(x) || ISLOWER(x) )

#define BIG(x,y) ( x > y ? x : y )

int main( )

{

char ch ; int d, a, b ;

printf ( "\nEnter any alphabet/character: " ) ;

scanf ( "%c", &ch ) ;

if ( ISUPPER ( ch ) == 1 )

printf ( "You entered a capital letter\n" ) ;

if ( ISLOWER ( ch ) == 1 )

printf ( "You entered a small case letter\n" ) ;

if ( ISALPHA ( ch ) != 1 )

printf ( "You entered character other than an alphabet\n" ) ;

printf ( "Enter any two numbers: " ) ;

scanf ( "%d%d", &a, &b ) ;

d = BIG ( a, b ) ;

printf ( "Bigger number is %d\n", d ) ;

return 0 ;

}

**(94)Write a program that interchanges elements at odd position with elements at even position in an array of 10 elements.**

**ANS:**

int main( )

{

int num[ ] = { 12, 4, 5, 1, 9, 13, 11, 19, 54, 34 } ;

int i, t ;

for ( i = 0 ; i <= 9 ; i = i + 2 )

{

t = num[ i ] ; num [ i ] = num [ i + 1 ] ; num [ i + 1 ] = t ;

}

for ( i = 0 ; i <= 9 ; i++ )

printf ( "%d\t", num[ i ] ) ;

return 0 ;

}

**(95) Write a program to copy the contents of a 5-element integer array into another array in reverse order.**

**ANS:**

int arr1[ 5 ], arr2[ 5 ], i, j ;

printf ( "\nEnter 5 elements of array:\n" ) ;

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

scanf ( "%d", &arr1[ i ] ) ;

for ( i = 0, j = 4 ; i <= 4 ; i++, j-- ) arr2[ j ] = arr1[ i ] ;

printf ( "Elements in reverse order:\n" ) ;

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

printf ( "%d\t", arr2[ i ] ) ;

return 0 ;

}

**(96)**

**calling functions**

# include<stdio.h>

void italy( ) ; void brazil( ) ; void argentina( ) ;

int main( )

{ printf ( "I am in main\n" ) ;

italy( ) ;

brazil( ) ;

argentina( ) ;

return 0 ; }

void italy( ) {

printf ( "I am in italy\n" ) ; }

void brazil( )

{ printf ( "I am in brazil\n" ) ;

} void argentina( ) {

printf ( "I am in argentina\n" ) ; }

The output : I am in main

I am in italy

I am in brazil

I am in argentina

**(97)Sending and receiving values between functions**

# include<stdio.h>

int calsum ( int x, int y, int z ) ;

int main( ) {

int a, b, c, sum ;

printf ( "Enter any three numbers " ) ;

scanf ( "%d %d %d", &a, &b, &c ) ;

sum = calsum ( a, b, c ) ;

printf ( "Sum = %d\n", sum ) ;

return 0 ; }

int calsum ( int x, int y, int z ) {

int d ; d = x + y + z ;

return ( d ) ; }

The output of the program..

Enter any three numbers 10 20 30

Sum = 60

**(98)When value of formal argument is changed**

# include<stdio.h>

void fun ( int ) ;

int main( ) {

int a = 30 ; fun ( a ) ;

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

return 0 ; }

void fun ( int b ) {

b = 60 ;

printf ( "%d\n", b ) ; }

The output : 60 30

**(99)Usage of library functions**

# include<stdio.h>

# include<stdlib.h>

int main( )

{ float a = 0.5 ;

float w, x, y, z ;

w = sin ( a ) ;

x = cos ( a ) ;

y = tan ( a ) ;

z = pow ( a, 2 ) ;

printf ( "%f %f %f %f\n", w, x, y, z ) ;

return 0 ; }

**(100) Return type of functions**

# include<stdio.h>

float square ( float ) ;

int main( ) {

float a, b ;

printf ( "Enter any number " ) ;

scanf ( "%f", &a ) ; b = square ( a ) ;

printf ( "Square of %f is %f\n", a, b ) ;

return 0 ; }

float square ( float x ) {

float y ; y = x \* x ;

return ( y ) ; }

sample runs of this program…

Enter any number 3

Square of 3 is 9.000000

Enter any number 1.5

Square of 1.5 is 2.250000

Enter any number 2.5

Square of 2.5 is 6.250000