// 1. Write a program that converts upper case to lower **HEADER FILE**

char upper\_to\_lower(char ch);

// 2. Write a program to calculate the area of Circle

float area\_of\_circle(float radius);

// 3. Given two character inputs, find number of characters between them. For example, if input is ‘A’ and ‘E’ the output is 3

int char\_between\_char(char ch\_1, char ch\_2);

// 4. Write a program to convert temperature from Celsius to Fahrenheit

float celsius\_to\_fahrenheit(float temperature);

// 5. Check if entered number is even or odd

unsigned int odd\_or\_even(int num);

// 6. Check if entered year is leap year or not

int check\_leap\_year(int year);

// 7. WAP to find power of 2^N using left shift operator

int pow\_of\_2(int raise);

// 8. Check if given input is a character or integer. (Use integer input within the range 0-9)

int char\_or\_int(char ch);

// 9. Generate two random numbers and find its sum (Hint: use rand() from stdlib.h)

unsigned int add\_rand(unsigned int x, unsigned int y);

// 10. Enter a 5 digit number and find the sum of its digits. For E.g. if int number = 12345; then sum = 15;

unsigned int sum\_of\_digits(unsigned int num);

// 11. Write a program to reverse the number. For E.g. If int number = 12345; then the output reverse = 54321;

unsigned int reverse\_of\_num(unsigned int num);

// 12. Write a program to count the occurrences of a digit in a number.

unsigned int count\_digit(unsigned int num, unsigned int digit);

// 13. WAP to check if a given number is a palindrome. For e.g. 12321, 56788765;

unsigned int check\_palindrome(unsigned int num);

// 14. Generate the first 'N' prime numbers. For Eg. If N=5 then 2,3,5,7,11

unsigned int is\_prime(unsigned int num);

// 15. Write a C program to display and find the sum of the series 1+11+111+....111 up to n.

// For eg. if n=4, the series is: 1+11+111+1111. Take the value of 'n' as input from the user

unsigned int sum\_of\_series(unsigned int num);

// 16. A number is called an Armstrong number if the sum of the cubes of the digits of the

// number is equal to the number. For example 153 = 1^3 + 5^3 + 3^3. Write a C

// program that asks the user to enter a number and returns if it is Armstrong or not.

unsigned int check\_if\_armstrong(unsigned int num);

// 17. Amicable numbers are found in pairs. A given pair of numbers is Amicable if the sum of the

// proper divisors (not including itself) of one number is equal to the other number and vice – versa.

// For example 220 & 284 are amicable numbers. First we find the proper divisors of 220,

// 220:1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110; 1+ 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284

// Now, 284: 1, 2, 4, 71, 142; 1 + 2 + 4 + 71 + 142 = 220

// Write a C program to check that the input pair of numbers is amicable

unsigned int check\_if\_amicable(unsigned int num\_1, unsigned int num\_2);

// 18. Write a menu driven program to read two integers & find their sum, difference & product.

unsigned int menu\_driven\_calculator(unsigned int num\_1, unsigned int num\_2, char operation);

// 19. Write a C program to calculate the volume of the following shapes: Cube, Cuboid,

// Sphere, Cylinder and Cone. Ask the user which one s/he wants to calculate, and take

// the appropriate required inputs. Then print the result. The input should be taken in the

// main function and calculations for every solid should be done in a separate function

// by passing appropriate arguments.

// Example: If the user chooses the option for cube, only one input is required i.e., the

// side. The volume is then calculated and printed.

// If the user chooses the option for cuboid, only three inputs are required i.e., length,

// breadth and height. The volume is then calculated and printed.

float volume\_of\_cube(float length);

float volume\_of\_cuboid(float length, float breadth, float height);

float volume\_of\_sphere(float radius);

float volume\_of\_cylinder(float radius, float height);

float volume\_of\_cone(float radius, float height);

void main\_menu(void);

// 20. An Electricity board charges the following rates for use of electricity.

// For the First 200 units : Rs 5 per unit

// For the next 100 units : Rs 7 per unit

// Beyond 300 units : Rs 10 Per unit.

// Write a C Program to read no of units consumed and print out total charge amount

float electricity\_bill(float units\_consumed);

// 21. WAP to convert a binary number to decimal and vice versa

unsigned int bin\_2\_dec(unsigned long int num);

unsigned long int dec\_2\_bin(unsigned int num);

// 22. Generate a sequence of numbers such that every number in the sequence is the sum of

// the previous three numbers. The first three numbers are 0,0,1.

int generate\_series(int arr[], int size);

// 23. WAP to print the following sketch with input N as number of rows

// \*\*\*\*

// \*\*\*

// \*\*

// \*

void print\_pattern\_1(int n);

// 24. Generate the following sequence using a single loop with N as number of rows

// \*

// \*\*

// \*\*\*

// \*\*\*\*

void print\_pattern\_2(int n);

// 25. Write a C program, which will print two digit numbers whose sum of both digit is

// nine. e.g. 18,27,36......

void generate\_series\_2(void);

// 26. Write a recursive function for calculating power of a number. Take base number and

// exponent from user.

int power(int base, int exponent);

// 27. Write a recursive function for calculating factorial of a number.

int factorial(int num);

// 28. Use recursive calls to evaluate F(x) = x + x3/3! + x5/5! + x7/7!+ …

float series\_evaluation(int x, int n);

// 29. Concatenate two integer values using macros

#define CONCATENATE(X, Y) X##Y

// 30. Find square of a number using macros.

#define SQUARE(X) X\*X

// 31. Write a menu driven program to display the mathematical functions like square root,

// natural log, log10x, power(x,n), Cos(x). (use math.h)

// Using Functions from math.h

void main\_menu\_math\_operations(void);

// 32. WAP to sort the given array in ascending and descending order.

// Function to print integer array

int print\_int\_array(int arr[], int arr\_size);

int ascending\_sort(int arr[], int arr\_size);

int descending\_sort(int arr[], int arr\_size);

// 33. Write a program to swap two numbers using function.

void swap\_by\_ref(int \*ptr\_1, int \*ptr\_2);

// 34. WAP to find minimum and maximum elements in a given array using the function

int min\_max\_array(int arr[], int arr\_size, int \*min, int \*max);

// 35. Write a C program that take 2 integer sets A[] and B[] as input and prints results of

// following set operations:

// I. A union B (Write function set\_union())

// II. A intersection B (Write function set\_intersection())

// III. A-B and B-A (Write function set\_difference())

int set\_union(int arr\_a[], int size\_a, int arr\_b[], int size\_b, int arr\_c[], int size\_c);

int set\_intersection(int arr\_a[], int size\_a, int arr\_b[], int size\_b, int arr\_c[], int size\_c);

int set\_difference(int arr\_a[], int size\_a, int arr\_b[], int size\_b, int arr\_c[], int size\_c);

// 36. WAP to store 10 numbers in an array. Remove the duplicate entries in the array.

int remove\_duplicate\_in\_array(int arr[], int arr\_size);

// 37. WAP to search for a given integer in an array using the linear search technique.

int linear\_search(int arr[], int arr\_size, int element);

// 38. WAP to search for a given integer in an array using the binary search technique

int binary\_search(int arr[], int arr\_size, int element);

// 39. Write a C program, that reads list of n integer and print sum of product of consecutive

// numbers. if n=7 and numbers are 4,5,2,5,6,4,7 then output is

// 4\*5+5\*2+2\*5+5\*6+6\*4+4\*7 = 122.

int sum\_of\_product(int arr[], int arr\_size);

// 40. WAP to read a string from the user and find the length of string.( Note: Do not use

// string.h header file)

int find\_string\_length(char str[]);

// 41. Input date, month and year from the user, and using switch case, display in

// worded format. e.g. input: d=16, m=7, y=1992

// Output: 16th July, 1992

void worded\_date(int day, int month, int year, char str[]);

// 42. Write a function that will scan a character string passed as an argument and convert

// all lower-case characters into their upper-case equivalents.

void lower\_to\_upper(char str[]);

// 43. Write a program to read a string from the user and reverse the string.( Note: Do not

// use string.h header file)

void reverse\_string(char str[]);

// 44. Write a program to check whether the given input string is palindrome.

int string\_palindrome(char \*str);

// 45. Write a program to concatenate two stings. .( Note: Do not use string.h header file)

void string\_concat(char \*str\_1, char \*str\_2);

// 46. WAP to construct 5 \* 5 matrix and display the contents. Use functions for

// construction and display of matrix.

// Create 2D matrix using double pointers

int \*\*create\_2d\_matrix(int rows, int columns);

// Initialize 2D matrix

int initialize\_2d\_matrix(int \*\*my\_matrix, int rows, int columns);

// Print 2D matrix

int print\_2d\_matrix(int \*\*my\_matrix, int rows, int columns);

// 47. Given a matrix of size NxN, find it’s Transpose.

int transpose\_2d\_matrix(int \*\*my\_matrix, int rows, int columns);

// 48. Given a matrix of size NxM. Find the sum of each column in the matrix.

int print\_column\_sum(int \*\*my\_matrix, int rows, int columns);

// 49. WAP program to check if given matrix is a sparse matrix. If the number of zeros in a

// matrix exceeds (n\*m)/2, where n, m is the dimension of the matrix, matrix is sparse

// matrix. Sparse matrix has more zero elements than nonzero elements.

int check\_if\_sparse(int rows, int columns, int my\_matrix[rows][columns]);

/\*

// 50. Write a program to store and print the roll no., name, age and marks of 10 students

// using structures.

#define NAME\_SIZE 20

#define MAX\_SIZE 5

struct \_student\_{

int roll\_no;

char name[NAME\_SIZE];

int age;

float marks;

};

typedef struct \_student\_ Student;

struct \_student\_array\_{

Student std[MAX\_SIZE];

int c\_size;

int t\_size;

};

typedef struct \_student\_array\_ Student\_Array;

// Function to Initialize Array

Student\_Array initialize\_array(int t\_size);

// Function to store data in to array

Student\_Array store\_data(Student\_Array std\_arr, Student std\_data);

// Function to print student array

void print\_data(Student\_Array std\_arr);

\*/

// 51. Write a program to add, subtract and multiply two complex numbers using structures.

struct \_complex\_num\_{

int real, img;

};

typedef struct \_complex\_num\_ Complex\_Num;

Complex\_Num add\_complex(Complex\_Num a, Complex\_Num b);

Complex\_Num sub\_complex(Complex\_Num a, Complex\_Num b);

Complex\_Num mul\_complex(Complex\_Num a, Complex\_Num b);

// 52. Take the price and quantity of items as an input. Write a C function to calculate the

// sum of the prices. Write another C function to calculate the discount according to the

// following rules:

// For total less than Rs.1000, discount is 5%.

// For total greater than Rs.1000 but less than Rs.5000, discount is 10%.

// For total greater than Rs.5000, discount is 15%.

// Write another function to print the individual item prices, total, discount and the final

// price. Example:

// If the prices are as follows:

// Item 1: 200

// Item 2: 400

// Item 3: 200

// Item 4: 10

// Item 5: 50

// And the quantities are:

// Item 1: 1

// Item 2: 1

// Item 3: 3

// Item 4: 5

// Item 5: 2

// Then you should print:

// Item Price Quantity Subtotal

// Item 1 200 1 200

// Item 2 400 1 400

// Item 3 200 3 600

// Item 4 10 5 50

// Item 5 50 2 100

// -------------------------------------------------

// TOTAL 1350

// Discount 10% -135

// -------------------------------------------------

// GRAND TOTAL 1215

// 53. Implement two player snake and ladder game with board size 10x10. Use 6 ladder and

// 7 snakes in the game. Use random function to roll the dice. After every move show

// the board to the user. Player who reaches 100 shall win the race. Note: Player1: user

// and Player2: computer.

// 54. WAP to store N integer values in an array (use malloc). Perform the following operations:

// i. Search an element using linear search

// ii. Find maximum and minimum value

// iii. Find the sum of all the elements of array

// iv. Find the sum of even and odd elements of array

// v. Check whether the array in palindrome

// vi. Find subset (x, y) such that it sums to M. Here (x, y) are two elements in an

// array and M is a whole number.

// vii. Write a function to deallocate the memory using free()

// 55. Write a program to store roll no., name, age and marks of N students dynamically.

// Perform the following operations

// i. Search student by roll no.

// ii. Search student by name

// iii. Return student details who have scored highest marks. (Assume only one

// student has scored highest marks)

// iv. Display the details of the students whose name begins with ‘A’.

// v. Deallocate all the memory before exiting the program

// 56. In an X-Y Plane there are N numbers of points at location (x,y). Each point has a

// weight associated with it. Define the structure for the above problem. Find the point

// with maximum weight. Assuming all points are connected, count the number of

// vertical and horizontal lines.

struct \_point\_{

int x, y;

int weight;

};

// 57. Write a simple program to display the contents of a file.

int disp\_file\_content(FILE \*ptr);

// 58. Write a program to copy the contents of one file to another.

int copy\_file\_content(FILE \*dest, FILE \*src);

// 59. Write a program to count number of characters, spaces tabs and lines in a file.

int char\_count\_in\_file(FILE \*fptr);

// 60. Read a file which contains one number per line. Check the number is odd or even and

// write to corresponding odd.txt or even.txt accordingly. Note: Numbers may not be

// single digit numbers.

int odd\_even\_sorting(FILE \*num\_ptr, FILE \*evn\_ptr, FILE \*odd\_ptr);

// 61. Write a program to create telephone directory. It should include Name, address and

// telephone number. It should have facilities for adding, deleting, editing and searching

// any number. Implement using text mode.

#define NAME\_SIZE 20

#define ADDR\_SIZE 50

struct \_contact\_{

char name[NAME\_SIZE];

char addr[ADDR\_SIZE];

char phone\_no[10];

};

typedef struct \_contact\_ Contact;

// Add record

int add\_record(FILE \*fptr, Contact \*contact);

// Delete record

// Edit record

// Search record

int search\_record(FILE \*fptr, long int phone\_no);

// 62. Write a program to create telephone directory. It should include Name, address and

// telephone number. It should have facilities for adding, deleting, editing and searching

// any number. Implement using binary mode.