Q1. WAP to check whether a given is Armstrong or not.

#include <stdio.h>

#include <math.h>

int main() {

int num = 153, sum = 0, temp, digits = 0, rem;

temp = num;

while (temp > 0) {

digits++;

temp /= 10;

}

temp = num;

while (temp > 0) {

rem = temp % 10;

sum += pow(rem, digits);

temp /= 10;

}

if (sum == num)

printf("Armstrong\n");

else

printf("Not Armstrong\n");

return 0;

}

Q2. WAP to read two integers and print their HCF (Highest Common

Factor).

#include <stdio.h>

int main() {

int a = 56, b = 98, temp;

while (b != 0) {

temp = b;

b = a % b;

a = temp;

}

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

return 0;

}

Q3. WAP to subtract two integers without using Minus (-) operator. (Hint

Bitwise operator)

#include <stdio.h>

int main() {

int a = 15, b = 7;

while (b != 0) {

int borrow = (~a) & b;

a = a ^ b;

b = borrow << 1;

}

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

return 0;

}

Q4. WAP to accept two integer numbers and swap them using 4

different methods in C language.

#include <stdio.h>

int main() {

//Using a third variable

int c = 10, d = 20, temp;

temp = c;

c = d;

d = temp;

printf("c = %d, d = %d\n"

, c, d);

//Without using a third variable (Arithmetic addition and

subtraction)

int e = 10, f = 20;

e = e + f;

f = e - f;

e = e - f;

printf("\ne = %d, f = %d\n"

, e, f);

//Without using a third variable (Arithmetic multiplication and

division)

int g = 10, h = 20;

g = g \* h;

h = g / h;

g = g / h;

printf("g = %d, h = %d\n"

, g, h);

//Using bitwise XOR operator

int a = 10, b = 20;

a = a ^ b;

b = a ^ b;

a = a ^ b;

printf("a = %d, b = %d\n"

, a, b);

return 0;

}

Q5. WAP to check whether number is Perfect Number or not.

#include <stdio.h>

int main() {

int num = 28, sum = 0, i;

for (i = 1; i < num; i++) {

if (num % i == 0) sum += i;

}

if (sum == num)

printf("Perfect Number\n");

else

printf("Not a Perfect Number\n");

return 0;

}

Q6. WAP to accept a coordinate point in an XY coordinate system and

determine in which quadrant the coordinate point lies.

#include <stdio.h>

int main() {

int x = 7, y = 9;

if (x > 0 && y > 0)

printf("The coordinate point (%d,%d) lies in the First

quadrant.\n", x, y);

else if (x < 0 && y > 0)

printf("The coordinate point (%d,%d) lies in the Second

quadrant.\n", x, y);

else if (x < 0 && y < 0)

printf("The coordinate point (%d,%d) lies in the Third

quadrant.\n", x, y);

else if (x > 0 && y < 0)

printf("The coordinate point (%d,%d) lies in the Fourth

quadrant.\n", x, y);

else

printf("The coordinate point (%d,%d) lies on the origin or an

axis.\n", x, y);

return 0;

}

Q7. WAP for Binary to Decimal conversion & Decimal to Binary for a

given number as per user’s choice.

#include <stdio.h>

#include <math.h>

int binaryToDecimal(int binary) {

int decimal = 0, base = 1;

while (binary > 0) {

int lastDigit = binary % 10;

decimal += lastDigit \* base;

base \*= 2;

binary /= 10;

}

return decimal;

}

void decimalToBinary(int decimal) {

int binary[32], i = 0;

while (decimal > 0) {

binary[i++] = decimal % 2;

decimal /= 2;

}

for (int j = i - 1; j >= 0; j--) {

printf("%d", binary[j]);

}

printf("\n");

}

int main() {

int choice, num;

printf("Enter 1 for Binary to Decimal, 2 for Decimal to Binary: ");

scanf("%d", &choice);

printf("Enter the number: ");

scanf("%d", &num);

if (choice == 1)

printf("Decimal: %d\n", binaryToDecimal(num));

else if (choice == 2) {

printf("Binary: ");

decimalToBinary(num);

}

return 0;

}

Q8. WAP to print below mentioned pattern:

1

01

101

0101

10101

#include <stdio.h>

int main() {

int n = 5;

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

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

if ((i + j) % 2 == 0)

printf("1");

else

printf("0");

}

printf("\n");

}

return 0;

}

Q9. WAP to print following Pyramid:

0 0

01 01

010 010

0101 0101

0101001010

#include <stdio.h>

int main() {

int n = 5;

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

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

printf("%d", j % 2);

}

for (int j = 1; j <= 2 \* (n - i); j++) {

printf(" ");

}

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

printf("%d", j % 2);

}

printf("\n");

}

return 0;

}

Assignment 2 (1D Array)

Q1. WAP to increase every student mark by 5 & then print the updated

array.

#include <stdio.h>

int main() {

int marks[] = {70, 80, 90, 85, 75};

int n = 5;

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

marks[i] += 5;

}

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

printf("%d ", marks[i]);

}

return 0;

}

Q2. WAP to print grade of students as per their marks given in an array.

(>=75—A grade, 74 to 60--B Grade, 59 to 40--C grade below 40--D grade).

#include <stdio.h>

int main() {

int marks[] = {78, 65, 55, 82, 39};

int n = 5;

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

if (marks[i] >= 75) {

printf("A ");

} else if (marks[i] >= 60) {

printf("B ");

} else if (marks[i] >= 40) {

printf("C ");

} else {

printf("D ");

}

}

return 0;

}

Q3. WAP to find who scored first “99” in an array marks.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 99, 39};

int n = 5;

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

if (marks[i] == 99) {

printf("First 99 found at index %d", i);

break;

}

}

return 0;

}

Q4. WAP to find Who & how many students have scored 99 in an array

Marks.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 99, 39};

int n = 5, count = 0;

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

if (marks[i] == 99) {

printf("99 found at index %d\n", i);

count++;

}

}

printf("Total students with 99: %d", count);

return 0;

}

Q5. WAP to find sum of all scores in Marks array.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 99, 39};

int n = 5, sum = 0;

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

sum += marks[i];

}

printf("Sum of all scores: %d", sum);

return 0;

}

Q6. WAP to find average score of the Marks array.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 99, 39};

int n = 5, sum = 0;

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

sum += marks[i];

}

float average = sum / (float)n;

printf("Average score: %.2f", average);

return 0;

}

Q7. WAP to check whether score is even or odd in an array.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 100, 39};

int n = 5;

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

if (marks[i] % 2 == 0) {

printf("%d is even\n", marks[i]);

} else {

printf("%d is odd\n", marks[i]);

}

}

return 0;

}

Q8. WAP to find maximum & minimum score in the Marks array.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 100, 39};

int n = 5;

int max = marks[0], min = marks[0];

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

if (marks[i] > max) {

max = marks[i];

}

if (marks[i] < min) {

min = marks[i];

}

}

printf("Maximum score: %d\n", max);

printf("Minimum score: %d", min);

return 0;

}

Q9. WAP to find a peak element which is not smaller than its neighbors.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 100, 39};

int n = 5;

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

if ((i == 0 || marks[i] >= marks[i - 1]) && (i == n - 1 || marks[i]

>= marks[i + 1])) {

printf("Peak element: %d", marks[i]);

break;

}

}

return 0;

}

Q10. WAP to count prime numbers in an array.

#include <stdio.h>

int main() {

int arr[] = {11, 14, 17, 20, 23};

int size = 5;

int count = 0;

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

int n = arr[i], isPrime = 1;

if (n <= 1) isPrime = 0;

for (int j = 2; j < n; j++) {

if (n % j == 0) {

isPrime = 0;

break;

}

}

if (isPrime) count++;

}

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

return 0;

}

Q11. WAP to implement Insert -Front, any position in between & end in

an array. Print the array before insert & after insert.

#include <stdio.h>

int main() {

int arr[100] = {10, 20, 30, 40, 50};

int size = 5;

int pos, val, i;

printf("Original Array: ");

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

printf("%d ", arr[i]);

}

printf("\n");

val = 5;

for (i = size; i > 0; i--) {

arr[i] = arr[i - 1];

}

arr[0] = val;

size++;

printf("After Insert at Front: ");

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

printf("%d ", arr[i]);

}

printf("\n");

pos = 3;

val = 25;

for (i = size; i > pos - 1; i--) {

arr[i] = arr[i - 1];

}

arr[pos - 1] = val;

size++;

printf("After Insert at Position %d: ", pos);

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

printf("%d ", arr[i]);

}

printf("\n");

val = 60;

arr[size] = val;

size++;

printf("After Insert at End: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q12. WAP to implement delete-Front, any position in between & end in

an array. Print the array before delete & after delete.

#include <stdio.h>

int main() {

int arr[100] = {10, 20, 30, 40, 50};

int size = 5, pos, i;

printf("Original Array: ");

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

printf("%d ", arr[i]);

}

printf("\n");

for (i = 0; i < size - 1; i++) {

arr[i] = arr[i + 1];

}

size--;

printf("After Delete at Front: ");

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

printf("%d ", arr[i]);

}

printf("\n");

pos = 2;

for (i = pos - 1; i < size - 1; i++) {

arr[i] = arr[i + 1];

}

size--;

printf("After Delete at Position %d: ", pos);

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

printf("%d ", arr[i]);

}

printf("\n");

size--;

printf("After Delete at End: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q13. Given an array, the task is to cyclically rotate the array clockwise

by one time.

Examples:

Input: arr[] = {1, 2, 3, 4, 5}

Output: arr[] = {5, 1, 2, 3, 4}

Input: arr[] = {2, 3, 4, 5, 1}

Output: {1, 2, 3, 4, 5}

#include <stdio.h>

int main() {

int arr[] = {1, 2, 3, 4, 5};

int size = 5, last, i;

last = arr[size - 1];

for (i = size - 1; i > 0; i--) {

arr[i] = arr[i - 1];

}

arr[0] = last;

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q14. Given an array of n integers. The task is to print the duplicates in

the given array.

If there are no duplicates then print -1.

Examples:

Input: {2, 10,10, 100, 2, 10, 11,2,11,2}

Output: 2 10 11

Input: {5, 40, 1, 40, 100000, 1, 5, 1}

Output: 5 40 1

#include <stdio.h>

int main() {

int arr[] = {2, 10, 10, 100, 2, 10, 11, 2, 11, 2};

int size = 10, i, j, flag = 0;

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

for (j = i + 1; j < size; j++) {

if (arr[i] == arr[j]) {

printf("%d ", arr[i]);

flag = 1;

break;

}

}

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

if (arr[i] == arr[j]) break;

}

}

if (flag == 0) printf("-1\n");

return 0;

}

Assignment 3 (Simple Calculator)

#include <stdio.h>

#include <math.h>

int main() {

int choice;

double num1, num2, result;

printf("Simple Calculator\n");

printf("Available Choices:\n");

printf("1. Addition\n");

printf("2. Subtraction\n");

printf("3. Multiplication\n");

printf("4. Division\n");

printf("5. Logarithmic Value\n");

printf("6. Square Root\n");

printf("\nType any number from above: ");

scanf("%d"

, &choice);

switch (choice) {

case 1:

printf("Enter two numbers: ");

scanf("%lf %lf"

, &num1, &num2);

result = num1 + num2;

printf("Result: %.2lf\n"

, result);

break;

case 2:

printf("Enter two numbers: ");

scanf("%lf %lf"

, &num1, &num2);

result = num1 - num2;

printf("Result: %.2lf\n"

, result);

break;

case 3:

printf("Enter two numbers: ");

scanf("%lf %lf"

, &num1, &num2);

result = num1 \* num2;

printf("Result: %.2lf\n"

, result);

break;

case 4:

printf("Enter two numbers: ");

scanf("%lf %lf"

, &num1, &num2);

if (num2 != 0) {

result = num1 / num2;

printf("Result: %.2lf\n"

, result);

} else {

printf("Error: Division by zero is not allowed.\n");

}

break;

case 5:

printf("Enter a number: ");

scanf("%lf"

, &num1);

result = log(num1);

printf("Logarithmic value: %.2lf\n"

, result);

break;

case 6:

printf("Enter a number: ");

scanf("%lf"

, &num1);

if (num1 >= 0) {

result = sqrt(num1);

printf("Square root: %.2lf\n"

, result);

} else {

printf("Error: Cannot calculate square root of a negative

number.\n");

}

break;

default:

printf("Invalid choice.\n");

break;

}

return 0;

}

Assignment 4 (Rock, Paper, Scissor)

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include <ctype.h>

int main() {

srand(time(0));

char \*aiChoice;

char plrChoice[15];

int random = rand() % 101;

if (random < 33)

aiChoice = "Rock";

else if (random < 66)

aiChoice = "Paper";

else

aiChoice = "Scissor";

printf("Computer chose: %s\n"

, aiChoice);

printf("Let's Play Rock, Paper, Scissor:\n");

printf("Type Rock, Paper, or Scissor: ");

scanf("%s"

, plrChoice);

strlwr(plrChoice);

if (strcmp(plrChoice, "rock") == 0) {

if (strcmp(aiChoice, "Paper") == 0) {

printf("\nYou Lost! AI chose Paper!!!\n");

}

else if (strcmp(aiChoice, "Scissor") == 0) {

printf("\nYou Won!! AI chose Scissor!!!\n");

}

else {

printf("\nIt's a Tie! You both chose Rock!!!\n");

}

}

else if (strcmp(plrChoice, "paper") == 0) {

if (strcmp(aiChoice, "Scissor") == 0) {

printf("\nYou Lost! AI chose Scissor!!!\n");

}

else if (strcmp(aiChoice, "Rock") == 0) {

printf("\nYou Won!! AI chose Rock!!!\n");

}

else {

printf("\nIt's a Tie! You both chose Paper!!!\n");

}

}

else if (strcmp(plrChoice, "scissor") == 0) {

if (strcmp(aiChoice, "Rock") == 0){

printf("\nYou Lost! AI chose Rock!!!\n");

}

else if (strcmp(aiChoice, "Paper") == 0){

printf("\nYou Won!! AI chose Paper!!!\n");

}

else {

printf("\nIt's a Tie! You both chose Scissor!!!\n");

}

}

else {

printf("\nInvalid input. Please type Rock, Paper, or

Scissor.\n");

}

return 0;

}

Assignment 5 (Hangman Game)

#include <stdio.h>

#include <string.h>

#include <ctype.h>

int main() {

char word[] = "coding";

char guessedWord[20];

int wordLength = strlen(word);

int attempts = 3;

int correctGuesses = 0;

char guess;

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

guessedWord[i] = '\_';

}

guessedWord[wordLength] = '\0';

printf("Welcome to Hangman!\n");

printf("You have %d attempts to guess the word.\n"

, attempts);

while (attempts > 0 && correctGuesses < wordLength) {

printf("Current word: %s\n"

, guessedWord);

printf("Enter your guess: ");

scanf(" %c"

, &guess);

guess = tolower(guess);

int found = 0;

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

if (word[i] == guess && guessedWord[i] == '\_') {

guessedWord[i] = guess;

correctGuesses++;

found = 1;

}

}

if (!found) {

attempts--;

printf("Wrong guess! Attempts remaining: %d\n"

,

attempts);

} else {

printf("Good guess!\n");

}

}

if (correctGuesses == wordLength) {

printf("Congratulations! You guessed the word: %s\n"

, word);

printf("The Man survives!\n");

} else {

printf("You ran out of attempts! The word was: %s\n"

, word);

printf("The Man is hanged!\n");

}

return 0;

}

Assignment 6 (Tic Tac Toe Game)

#include <stdio.h>

char board[3][3];

char player = 'X';

void initializeBoard() {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

board[i][j] = ' ';

}

}

}

void printBoard() {

printf("\n");

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

printf(" %c "

, board[i][j]);

if (j < 2) printf("|");

}

if (i < 2) printf("\n---|---|---\n");

}

printf("\n");

}

int checkWin() {

for (int i = 0; i < 3; i++) {

if (board[i][0] == player && board[i][1] == player &&

board[i][2] == player) return 1;

if (board[0][i] == player && board[1][i] == player &&

board[2][i] == player) return 1;

}

if (board[0][0] == player && board[1][1] == player && board[2][2]

== player) return 1;

if (board[0][2] == player && board[1][1] == player && board[2][0]

== player) return 1;

return 0;

}

int checkDraw() {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

if (board[i][j] == ' ') return 0;

}

}

return 1;

}

void switchPlayer() {

player = (player == 'X') ? 'O' : 'X';

}

void makeMove() {

int row, col;

while (1) {

printf("Player %c, enter your move (row and column): ",

player);

scanf("%d %d"

, &row, &col);

if (row >= 1 && row <= 3 && col >= 1 && col <= 3 &&

board[row-1][col-1] == ' ') {

board[row-1][col-1] = player;

break;

} else {

printf("Invalid move! Try again.\n");

}

}

}

int main() {

initializeBoard();

while (1) {

printBoard();

makeMove();

if (checkWin()) {

printBoard();

printf("Player %c wins!\n"

, player);

break;

}

if (checkDraw()) {

printBoard();

printf("It's a draw!\n");

break;

}

switchPlayer();

}

return 0;

}