***ALL PROGRAMS OF 1ST SEMESTER OF PF (PROGRAMMING FUNDAMENTALS)***

**Problem 1:**

**A cashier has currency notes of denominations 10, 50, and 100. If the amount to be withdrawn is input**

**through the keyboard in hundreds, find the total number of currency notes of each denomination the**

**cashier will have to give to the withdrawer. Write a C++ program of the details mentioned above.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int amount, tens, fiftys, hundreds;

cout << "\t Expected Output:"<<endl;

cout << "Example 1:" << endl; //first time input

cout << "Enter amount in hundred: ";

cin >> amount;

tens = amount / 10;

fiftys = amount / 50;

hundreds = amount / 100;

cout << "Currency note in denomination 10 are : " << tens << endl;

cout << "Currency note in denomination 50 are : " << fiftys <<endl;

cout << "Currency note in denomination 100 are : " << hundreds <<endl;

cout << "Example 2:" << endl; //second time input

cout << "Enter amount in hundred: ";

cin >> amount;

tens = amount / 10;

fiftys = amount / 50;

hundreds = amount / 100;

cout << "Currency note in denomination 10 are : " << tens << endl;

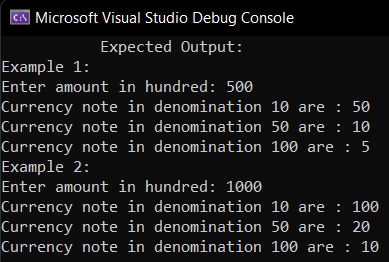
cout << "Currency note in denomination 50 are : " << fiftys << endl;

cout << "Currency note in denomination 100 are : " << hundreds << endl;

return 0;

}

**OUTPUT:**

****

**Problem 2:**

**Write a C++ program that calculates the monthly salary of an employee. The salary is calculated after**

**the deduction of the following taxes.**

**Income Tax: 10%**

**Global Life Insurance: 2.75%**

**Pension Plan: 2.3%**

**Health Insurance: 200.0 rupees**

**Your program should prompt the user to input the gross amount and your program should show each**

**tax and net salary (after tax deduction). Your output should be in the same format (not the same values)**

**as shown in the examples below.**

**Expected output:**

**Example 1:**

**Enter Gross amount of employee(RS. ): 100000**

**Income Tax (RS.): 10000**

**Global Life Insurance(RS.): 2750**

**Pension Plan (RS.):2300**

**Health Insurance: 200.0 rupees**

**Net Salary (after tax deduction) (Rs. ): 84750**

**Code:**

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

double monthly\_salary, income\_tax, G\_L\_insurance, pension,net\_amount;

float health\_insurance = 200.0;

cout << "\t Expected Output:" << endl;

cout << "Example 1:" << endl;

cout << fixed << setprecision(1);

cout << "Enter Gross amount of employee(RS. ): ";

cin >> monthly\_salary;

income\_tax = monthly\_salary \* 0.1;

G\_L\_insurance = monthly\_salary \* 0.0275;

pension = monthly\_salary \* 0.023;

net\_amount = monthly\_salary - income\_tax - G\_L\_insurance - pension - health\_insurance;

cout << "Income Tax (RS.): " << income\_tax<<endl;

cout << "Global Life Insurance(RS.): " << G\_L\_insurance << endl;

cout << "Pension Plan (RS.): " << pension << endl;

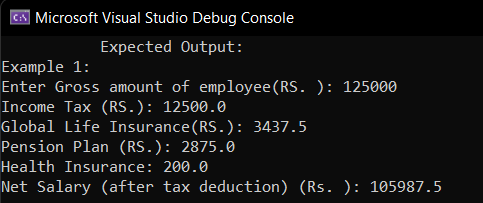
cout << "Health Insurance: " << health\_insurance << endl;

cout << "Net Salary (after tax deduction) (Rs. ): " << net\_amount << endl;

return 0;

}

**Output:**

****

**Problem 3:**

**Write a C++ program that takes a 5-digit number in variable X. Your program should store the reverse of the entered number into variable Y. Your program also prints it on screen.**

**Note: you are not supposed to take 5 separate numbers, but you have to take one number of 5 digits in a single variable. Also, store its reverse in a single variable.**

**Output:**

**Example 1:**

**Enter 5 Digit number in Variable X: 57391**

**5 Digit number in reverse order in Variable Y is: 19375**

**Code:**

#include <iostream>

using namespace std;

int main() {

int x, y = 0; // x for input, y for reversed number

// Input a 5-digit number

cout << "Enter 5-digit number in variable X: ";

cin >> x;

y = (x % 10) \* 10000;

x =x / 10;

y = y + (x % 10) \* 1000;

x = x / 10;

y = y + (x % 10) \* 100;

x = x / 10;

y = y + (x % 10) \* 10;

x = x / 10;

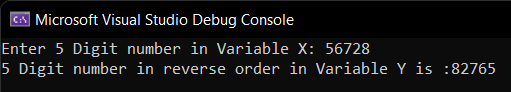
y = y + (x % 10);

cout << " 5 digit number in reverse order in variable Y is :" << y << endl;

return 0;

}

**Output:**

****

**Problem 4:**

**Write a program that takes marks of 5 courses as input, of 5 students and outputs the students who got the highest aggregate without utilizing any loops.**

**Code:**

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

int a1, a2, a3, a4, a5, rolla, A;

int b1, b2, b3, b4, b5, rollb, B;

int c1, c2, c3, c4, c5, rollc, C;

int d1, d2, d3, d4, d5, rolld, D;

int e1, e2, e3, e4, e5, rolle, E;

cout << "\t\tRoll#\t C1\tC2\tC3\tC4\tC5" << endl;

cout << "Input:" << endl;

cout << "\t\t";

cin >> rolla;

cin >> a1 >> a2 >> a3 >> a4 >> a5;

cout << endl;

cout << "\t\t";

cin >> rollb;

cin >> b1 >> b2 >> b3 >> b4 >> b5;

cout << endl;

cout << "\t\t";

cin >> rollc;

cin >> c1 >> c2 >> c3 >> c4 >> c5;

cout << endl;

cout << "\t\t";

cin >> rolld;

cin >> d1 >> d2 >> d3 >> d4 >> d5;

cout << endl;

cout << "\t\t";

cin >> rolle;

cin >> e1 >> e2 >> e3 >> e4 >> e5;

cout << "Sample Output:";

cout << endl;

A = a1 + a2 + a3 + a4 + a5;

B = b1 + b2 + b3 + b4 + b5;

C = c1 + c2 + c3 + c4 + c5;

D = d1 + d2 + d3 + d4 + d5;

E = e1 + e2 + e3 + e4 + e5;

if ((A > B && A > C) && (A > D && A > E))

cout << rolla << " has highest aggregrate of " << A;

else if ((B > A && B > C) && (B > D && B > E))

cout << rollb << " has highest aggregrate of " << B;

else if ((C > A && C > B) && (C > D && C > E))

cout << rollc << " has highest aggregrate of " << C;

else if ((D > A && D > B) && (D > C && D > E))

cout << rolld << " has highest aggregrate of " << D;

else if ((E > A && E > B) && (E > C && E > D))

cout << rolle << " has highest aggregrate of " << E;

else if ((A == B && A > C) && (A > D && A > E))

cout << rolla << " and " << rollb << " has highest aggregrate of " << A;

else if ((A == C && A > B) && (A > D && A > E))

cout << rolla << " and " << rollc << " has highest aggregrate of " << A;

else if ((A == D && A > B) && (A > C && A > E))

cout << rolla << " and " << rolld << " has highest aggregrate of " << A;

else if ((A == E && A > B) && (A > C && A > D))

cout << rolla << " and " << rolle << " has highest aggregrate of " << A;

else if ((B == C && B > A) && (B > D && B > E))

cout << rollb << " and " << rollc << " has highest aggregrate of " << B;

else if ((B == D && B > A) && (A > C && A > E))

cout << rollb << " and " << rolld << " has highest aggregrate of " << B;

else if ((B == E && B > A) && (B > C && B > D))

cout << rollb << " and " << rolle << " has highest aggregrate of " << B;

else if ((C == D && C > A) && (C > B && C > E))

cout << rollc << " and " << rolld << " has highest aggregrate of " << C;

else if ((C == E && C > A) && (C > B && C > D))

cout << rollc << " and " << rolle << " has highest aggregrate of " << C;

else if ((D == E && D > A) && (D > B && D > C))

cout << rolld << " and " << rolle << " has highest aggregrate of " << D;

else if (A == B && A == C && A > D && A > E)

cout << rolla << ", " << rollb << ", and " << rollc << " have the highest aggregate of " << A;

else if (A == B && A == D && A > C && A > E)

cout << rolla << ", " << rollb << ", and " << rolld << " have the highest aggregate of " << A;

else if (A == B && A == E && A > C && A > D)

cout << rolla << ", " << rollb << ", and " << rolle << " have the highest aggregate of " << A;

else if (A == C && A == D && A > B && A > E)

cout << rolla << ", " << rollc << ", and " << rolld << " have the highest aggregate of " << A;

else if (A == C && A == E && A > B && A > D)

cout << rolla << ", " << rollc << ", and " << rolle << " have the highest aggregate of " << A;

else if (A == D && A == E && A > B && A > C)

cout << rolla << ", " << rolld << ", and " << rolle << " have the highest aggregate of " << A;

else if (B == C && B == D && B > A && B > E)

cout << rollb << ", " << rollc << ", and " << rolld << " have the highest aggregate of " << B;

else if (B == C && B == E && B > A && B > D)

cout << rollb << ", " << rollc << ", and " << rolle << " have the highest aggregate of " << B;

else if (B == D && B == E && B > A && B > C)

cout << rollb << ", " << rolld << ", and " << rolle << " have the highest aggregate of " << B;

else if (C == D && C == E && C > A && C > B)

cout << rollc << ", " << rolld << ", and " << rolle << " have the highest aggregate of " << C;

else if (A == B && A == C && A == D && A > E)

cout << rolla << ", " << rollb << ", " << rollc << ", and " << rolld << " have the highest aggregate of " << A;

else if (A == B && A == C && A == E && A > D)

cout << rolla << ", " << rollb << ", " << rollc << ", and " << rolle << " have the highest aggregate of " << A;

else if (A == B && A == D && A == E && A > C)

cout << rolla << ", " << rollb << ", " << rolld << ", and " << rolle << " have the highest aggregate of " << A;

else if (A == C && A == D && A == E && A > B)

cout << rolla << ", " << rollc << ", " << rolld << ", and " << rolle << " have the highest aggregate of " << A;

else if (B == C && B == D && B == E && B > A)

cout << rollb << ", " << rollc << ", " << rolld << ", and " << rolle << " have the highest aggregate of " << B;

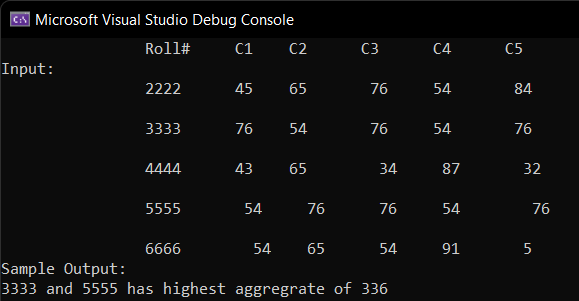
else if (A == B && A == C && A == D && A == E)

cout << rolla << ", " << rollb << ", " << rollc << ", " << rolld << ", and " << rolle << " all have the same highest aggregate of " << A;

return 0;

}

**Output:**



**Problem 5:**

**Write and run a program that plays the game of “Rock, paper, scissors.” In this game, two**

**players simultaneously say (or display a hand symbol representing) either “rock,” “paper,” or**

**“scissors.” The winner is the one whose choice dominates the other.**

**The rules are: paper dominates (wraps) rock, rock dominates (breaks) scissors, and scissors**

**dominate (cut) paper.**

**You can use 1=rock, 2=paper, 3=scissor**

**Code:(If total turns are 10)**

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

int a, b;

int c = 0, d = 0;

cout << "1=rock\t\t2=paper\t\t3=scissor" << endl;

cout << "You have total 10 turns.The player who wins most turns from 10 will win the battle." << endl << endl;

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

{

cout << "player 1 turn: ";

cin >> a;

cout << "player 2 turn: ";

cin >> b;

if ((a == 1 && b == 1) || (a == 2 && b == 2) || (a == 3 && b == 3))

cout << "Draw" << endl;

else if (a == 1 && b == 2)

{

cout << "Player 2 wins" << endl;

d = d + 1;

}

else if (a == 1 && b == 3)

{

cout << "Player 1 wins" << endl;

c = c + 1;

}

else if (a == 2 && b == 1)

{

cout << "Player 1 wins" << endl;

c = c + 1;

}

else if (a == 2 && b == 3)

{

cout << "Player 2 wins" << endl;

d = d + 1;

}

else if (a == 3 && b == 1)

{

cout << "Player 2 wins" << endl;

d = d + 1;

}

else if (a == 3 && b == 2)

{

cout << "Player 1 wins" << endl;

c = c + 1;

}

else

{

cout << "Invalid number" << endl;

i--;

}

cout << endl;

}

cout << "-----------------"<<endl;

if (c > d)

cout << "Overall Player 1 wins";

else if (d > c)

cout << "Overall Player 2 wins";

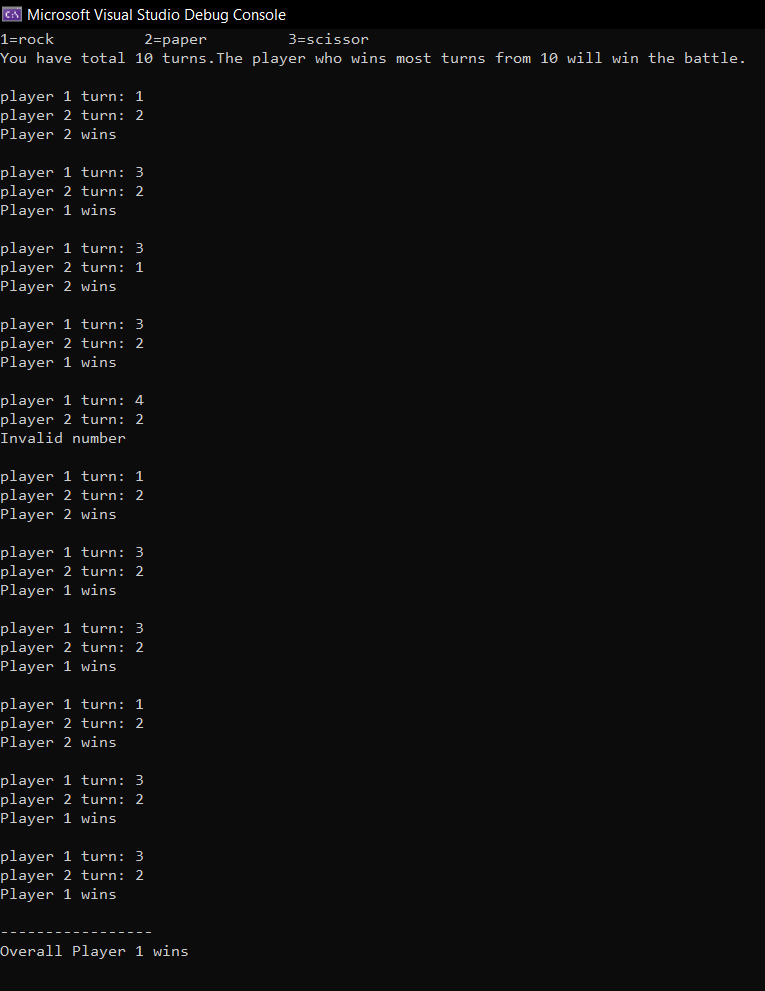
else

cout << "Draw";

return 0;

}

**Output:**

****

**Code:(If 10 is the limit to win for a player)**

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

int a, b;

int c = 0, d = 0;

cout << "1=rock\t\t2=paper\t\t3=scissor" << endl;

cout << "The player who do 10 points will win the match." << endl << endl;

for (;c<=10&&d<=10 ;)

{

cout << "player 1 turn: ";

cin >> a;

cout << "player 2 turn: ";

cin >> b;

if ((a == 1 && b == 1) || (a == 2 && b == 2) || (a == 3 && b == 3))

{

cout << "Draw" << endl;

}

else if (a == 1 && b == 2)

{

cout << "Player 2 wins" << endl;

d = d + 1;

}

else if (a == 1 && b == 3)

{

cout << "Player 1 wins" << endl;

c = c + 1;

}

else if (a == 2 && b == 1)

{

cout << "Player 1 wins" << endl;

c = c + 1;

}

else if (a == 2 && b == 3)

{

cout << "Player 2 wins" << endl;

d = d + 1;

}

else if (a == 3 && b == 1)

{

cout << "Player 2 wins" << endl;

d = d + 1;

}

else if (a == 3 && b == 2)

{

cout << "Player 1 wins" << endl;

c = c + 1;

}

else

{

cout << "Invalid number" << endl;

}

if (c == 10 || d == 10)

{

break;

}

cout << endl;

}

cout << "-----------------"<<endl;

if (c > d)

cout << "Overall Player 1 wins";

else

cout << "Overall Player 2 wins";

return 0;

**}**

**Problem 6:**

**Write a program to find the largest digit in a given number. The program should use a function to perform the task. You must define a function that takes the number as input and returns the largest digit. Additionally, use a while loop within the function to process the number.**

**Output:**

**Sample Input:**

**87239**

**Sample Output:**

**9**

Code:

#include<iostream>

using namespace std;

int largest(int num)

{

int last, larger = -10;

if (num < 0) //handle negative numbers

{

while (num != 0)

{

last = num % 10;

if (last > larger)

{

larger = last;

}

num = num / 10;

}

return larger;

}

else //handle positive numbers

{

while (num != 0)

{

last = num % 10;

if (last > larger)

larger = last;

num = num / 10;

}

return larger;

}

}

int main()

{

int num,larger;

cout << "Enter a number: ";

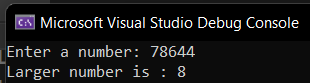
cin >> num;

larger=largest(num);

cout << "Larger number = "<<larger;

}

Output:



**Problem 7:**

**Create a program that prints the first n numbers in the Fibonacci sequence. Implement this by using a function that handles the logic. The function should take n as a parameter and use a for loop to calculate and print the sequence.**

**Note: The Fibonacci sequence is a series where each number is the sum of the two preceding ones, starting from 0 and 1. The sequence looks like this: 0, 1, 1, 2, 3, 5, 8, 13, and so on.**

**Sample Input:**

**5**

**Sample Output:**

**0 1 1 2 3**

Code:

#include<iostream>

using namespace std;

void fibonacci(int );

int main()

{

int num;

cout << "Enter the limit for fibonacci sequence: ";

cin >> num;

fibonacci(num);

}

void fibonacci(int n)

{

int sum = 0, a = 0, b = 1;

if (n == 0)

{

cout << "Invalid number. Please Enter a number greater than 0";

}

else if (n == 1)

{

cout << "0";

}

else if (n == 2)

{

cout << "0,1";

}

else

{

cout << "0,1,";

for (int i = 3; i <= n; i++)

{

sum = a + b;

a = b;

b = sum;

cout << sum << ",";

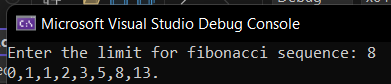
}

cout << "\b.";

}

}

Output:



**Problem 8:**

**Write a C++ program that takes two numbers from the user, start and end, and displays**

**all the prime numbers in that range. Implement this by using a function that handles the**

**logic.**

**Sample Input:**

**Enter the start number: 10**

**Enter the end number: 30**

**Sample Output:**

**The prime numbers between 10 and 30 are: 11, 13, 17, 19, 23, 29.**

Code:

#include<iostream>

using namespace std;

int prime(int start, int end)

{

int c;

for (int i = start + 1; i < end; i++)

{

c = 0;

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

{

if (i % j == 0)

{

c++;

}

}

if (c == 2)

cout << i << ",";

}

cout << "\b.";

return 0;

}

int main()

{

int start, end;

cout << "Enter start number: ";

cin >> start;

cout << "Enter end number: ";

cin >> end;

if(start<end)

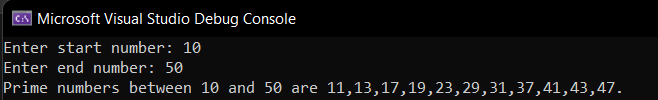
cout << "Prime numbers between " << start << " and " << end << " are ";

prime(start, end);

return 0;

}

Output:



**Problem 9:**

**Write a C++ program that prints a diamond shape made of characters. The program**

**should input an odd integer n, representing the diamond’s height. The diamond should**

**be centered and consist of characters starting from 'A' at the top and increasing until the**

**character corresponding to the n-th letter of the alphabet at the widest part, then**

**decreasing back to 'A'. Implement this by using a function that handles the logic for**

**generating the diamond.**

**Sample Input:**

**Enter an odd integer for the height of the diamond: 5**

**Sample Output:**

**A**

**ABA**

**ABCBA**

**ABCDCBA**

**ABCDEDCBA**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int num;

cout << "Enter odd number for the height of the diamond: ";

cin >> num;

char ch;

for (int i = 0; i < num; i++)

{

ch = 'A';

for (int j = 1; j < num - i; j++)

{

cout << " ";

}

for (int k = 0; k <= i; k++)

{

cout << ch;

ch++;

}

ch =ch-2;

for (int k = 0; k < i; k++)

{

cout << ch;

ch--;

}

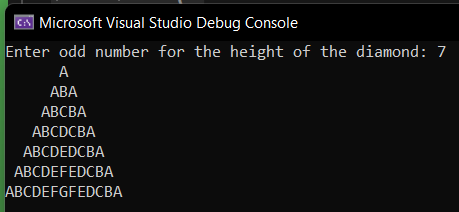
cout << endl;

}

return 0;

}

Output



**Problem 10:**

**Create a simple banking system simulator with a menu-driven interface that allows users**

**to perform the following operations:**

**1. Deposit Money**

**2. Withdraw Money**

**3. Check Balance**

**4. Check if the Balance is Below a Certain Amount**

**5. Exit**

**Details:**

**1. Deposit Money:**

**○ Implement two functions to deposit money:**

**■ void deposit(double amount): This function accepts a double**

**value representing the amount to deposit. It adds this amount to**

**the global balance variable.**

**■ void deposit(double amount, const string& transactionType):**

**This overloaded function also accepts a double amount and a**

**string representing the transaction type (e.g., "bonus"). If the**

**transaction type is "bonus", a 10% bonus is added to the deposit**

**amount.**

**2. Withdraw Money:**

**○ Implement two functions to withdraw money:**

**■ void withdraw(double amount): This function accepts a double**

**value representing the amount to withdraw. It subtracts this**

**amount from the global balance variable if sufficient funds are**

**available. Otherwise, it indicates insufficient funds.**

**■ void withdraw(double amount, const string&**

**transactionType): This overloaded function accepts a double**

**amount and a string representing the transaction type (e.g., "fee").**

**If the transaction type is "fee", a $2 fee is added to the withdrawal**

**amount before subtracting from the balance.**

**3. Check Balance:**

**○ Implement a function:**

**■ void checkBalance(): This function displays the current value of**

**the global balance variable.**

**4. Check if the Balance is Below a Certain Amount:**

**○ Implement a function:**

**■ void checkBalance(double amount): This function accepts a**

**double value representing a threshold amount and checks if the**

**current balance is below this threshold. It displays whether the**

**balance is above or below the specified amount.**

**5. Global Variables:**

**○ double balance = 0.0;: A global variable to keep track of the current**

**account balance.**

**○ int transactionCount = 0;: A global variable to count the number of**

**transactions performed.**

**Example Output:**

**1. Deposit Money**

**2. Withdraw Money**

**3. Check Balance**

**4. Check if the Balance is Below a Certain Amount**

**5. Exit**

**Enter your choice: 1**

**Enter amount to deposit: 150.75**

**Deposit successful. Current balance: 150.75**

**1. Deposit Money**

**2. Withdraw Money**

**3. Check Balance**

**4. Check if the Balance is Below a Certain Amount**

**5. Exit**

**Enter your choice: 3**

**Current balance: 150.75**

**1. Deposit Money**

**2. Withdraw Money**

**3. Check Balance**

**4. Check if the Balance is Below a Certain Amount**

**5. Exit**

**Enter your choice: 4**

**Enter the amount to compare: 100.00**

**Result: Balance is above 100.00**

**1. Deposit Money**

**2. Withdraw Money**

**3. Check Balance**

**4. Check if the Balance is Below a Certain Amount**

**5. Exit**

**Enter your choice: 5**

**Exiting program...**

Code:

#include<iostream>

using namespace std;

double balance = 0.0;

int transactioncount = 0;

void deposit(double amount)

{

balance = balance + amount;

cout << "Deposit Succesful.Current balance is " << balance << endl;

transactioncount++;

}

void deposit(double amount, const string& transactionType)

{

double y;

cout << "Enter transaction type: ";

if (transactionType == "bonus")

{

balance = balance + amount;

y = balance \* 0.1;

balance = balance + y;

cout << "Deposit Succesful.Current balance is " << balance << endl;

transactioncount++;

}

else

cout << "Invalid Transaction Type";

}

void withdraw(double amount)

{

if (balance >= amount)

{

balance = balance - amount;

cout << "Your remaining balance is " << balance << endl;

}

else

{

cout << "Insufficent balance" << endl;

cout << "Your balance is: " << balance;

}

transactioncount++;

}

void withdraw(double amount, const string& transactionType)

{

double y;

cout << "Enter transaction type: ";

if (transactionType == "fee")

{

amount = amount + 2;

balance = balance - amount;

cout << "Your remaining balance is " << balance << endl;

transactioncount++;

}

else

cout << "Invalid Transaction Type";

}

void checkbalance()

{

cout << "Current balance is " << balance << endl;

}

void checkbalance(double amount)

{

if (balance > amount)

{

cout << "Balance is above " << amount << endl;

}

else if (balance < amount)

{

cout << "Balance is below " << amount << endl;

}

else

{

cout << "Your balance is equal to the " << amount << endl;

}

}

int main()

{

int choice, i = 1;

double amount;

while (i == 1)

{

cout << "1. Deposit Money\n2. Withdraw Money\n3. Check Balance\n4. Check if balance is below a certain"

<< " amount\n5. Exit" << endl << endl;

cout << "Enter your choice: ";

cin >> choice;

if (choice == 1)

{

cout << "Enter the amount to deposit: ";

cin >> amount;

deposit(amount);

cout << endl;

system("pause");

cout << endl;

}

else if (choice == 2)

{

cout << "Enter the amount to withdraw: ";

cin >> amount;

withdraw(amount);

cout << endl;

system("pause");

cout << endl;

}

else if (choice == 3)

{

;

checkbalance();

cout << endl;

system("pause");

cout << endl;

}

else if (choice == 4)

{

cout << "Enter the amount to compare: ";

cin >> amount;

checkbalance(amount);

cout << endl;

system("pause");

cout << endl;

}

else if (choice == 5)

{

cout << "Exiting Program";

break;

}

else

{

cout << "Invalid Choice";

break;

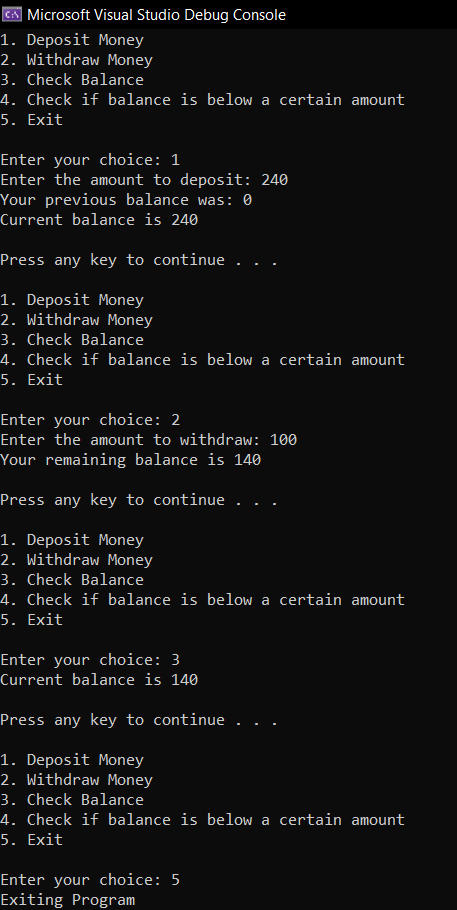
}

}

return 0;

}

**Output:**



**Problem 11:**

**Write a program that asks the user to enter two numbers and prints their sum, difference,**

**product, and quotient**.

**Code:**

#include<iostream>

using namespace std;

int main(){

int a, b;

cout << "First integer is: ";

cin >> a;

cout << "Second integer is: ";

cin >> b;

cout << "a +b = " << a + b << endl;

cout << "a - b = " << a - b << endl;

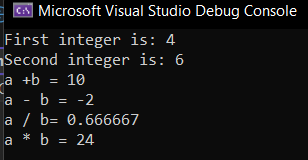
cout << "a / b= " << (float) a / b << endl;

cout << "a \* b = " << a \* b << endl;

return 0;

}

**Output:**



**Problem 12:**

**Write a program that asks the user to enter a weight in kilograms and converts it to pounds. (1kg = 2.20462 lbs).**

**Code:**

#include<iostream>

using namespace std;

int main() {

double kg, pound;

cout << "Enter weight in kilogram ";

cin >> kg;

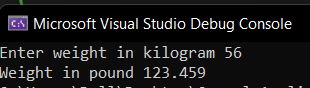
pound = kg \* 2.20462;

cout << "Weight in pound " << pound;

return 0;

}

**Output:**



**Problem:13**

**Swap the values of two variables :**

* **Without third variable**
* **With third variable**

**Code:**

#include<iostream>

using namespace std;

int main()

{

//swapping values without third variable

int a = 5, b = 10;

cout << "Before swapping the values are a= " << a << " and b= " << b<<endl;

b = a + b;

a = b - a;

b = b - a;

cout << "After swapping the values are a= " << a << " and b= " << b;

//swapping values with third variable

int a = 5, b = 10,temp;

cout << "Before swapping the values are a= " << a << " and b= " << b << endl;

temp = a;

a = b;

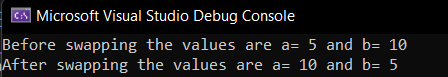
b = temp;

cout << "After swapping the values are a= " << a << " and b= " << b;

return 0;

}

**Output:**



**Problem 14**

**Write a program that takes total and obtained marks of a student in 3 subjects EE, Maths and FCP. Calculate overall and individual percentage of a subjects.**

**Code:**

#include <iostream>

using namespace std;

int main() {

// Variables to hold obtained and total marks for each subject

float obtained\_fcp, total\_fcp;

float obtained\_math, total\_math;

float obtained\_ee, total\_ee;

// Input for obtained and total marks in three subjects

cout << "Enter total marks in FCP: ";

cin >> total\_fcp;

cout << "Enter total marks in Math: ";

cin >> total\_math;

cout << "Enter total marks in EE: ";

cin >> total\_ee;

cout <<endl<< "Enter obtained marks in FCP: ";

cin >> obtained\_fcp;

cout << "Enter obtained marks in Math: ";

cin >> obtained\_math;

cout << "Enter obtained marks in EE: ";

cin >> obtained\_ee;

float percentage\_fcp = (obtained\_fcp / total\_fcp) \* 100;

float percentage\_math = (obtained\_math / total\_math) \* 100;

float percentage\_ee = (obtained\_ee / total\_ee) \* 100;

float overall\_obtained = obtained\_fcp + obtained\_math + obtained\_ee;

float overall\_total = total\_fcp + total\_math + total\_ee;

float overall\_percentage = (overall\_obtained / overall\_total) \* 100;

cout << "\nPercentage in FCP: " << percentage\_fcp << "%" << endl;

cout << "Percentage in Math: " << percentage\_math << "%" << endl;

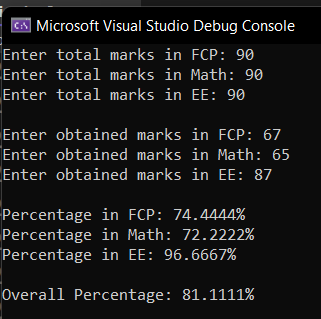
cout << "Percentage in EE: " << percentage\_ee << "%" << endl;

cout << "\nOverall Percentage: " << overall\_percentage << "%" << endl;

return 0;

}

**Output:**



**Problem 15**

**Write a C++ code to get three numbers from the user and find their average.**

**Code:**

#include <iostream>

using namespace std;

int main() {

float num1, num2, num3, average;

cout << "Enter the first number: ";

cin >> num1;

cout << "Enter the second number: ";

cin >> num2;

cout << "Enter the third number: ";

cin >> num3;

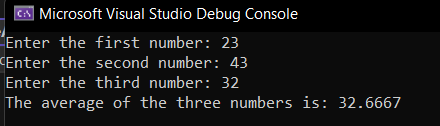
average = (num1 + num2 + num3) / 3;

cout << "The average of the three numbers is: " << average << endl;

return 0;

}

**Output:**



**Problem 16**

**Write a C++ program to get distance and time from the user and calculate the velocity.**

**Hint: Velocity = distance / time**

**Code:**

#include <iostream>

using namespace std;

int main() {

float distance, time, velocity;

cout << "Enter the distance (in meters): ";

cin >> distance;

cout << "Enter the time (in seconds): ";

cin >> time;

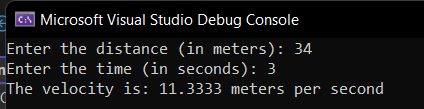
velocity = distance / time;

cout << "The velocity is: " << velocity << " meters per second" << endl;

return 0;

}

**Output:**



**Problem 16:**

**Write a program to create a simple calculator. Your program should allow users to enter two**

**integers and perform arithmetic operations using assignment operators. Implement addition,**

**subtraction, multiplication, division, and modulus operations. Display the results of each**

**operation.**

**Code:**

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

int i, j;

cout << "Enter two integers: ";

cin >> i >> j;

cout << fixed << setprecision(1);

cout << i << " + " << j << " = " << i + j << endl;

cout << i << " - " << j << " = " << i - j << endl;

cout << i << " \* " << j << " = " << i \* j << endl;

cout << i << " / " << j << " = " << i/float(j) << endl;

cout << i << " % " << j << " = " << i % j << endl;

return 0;

}

**Problem 17:**

**Write a program that compares two integers entered by the user using relational operators.**

**Display the results of each comparison operation (==, !=, >, <, >=, <=).**

**Code:**

#include <iostream>

using namespace std;

int main() {

int a, b;

bool result;

cout << "Enter two integers: ";

cin >> a >> b;

result = (a == b);

cout << a << " == " << b << " : " << result << endl;

result = (a != b);

cout << a << " != " << b << " : " << result << endl;

result = (a > b);

cout << a << " > " << b << " : " << result << endl;

result = (a < b);

cout << a << " < " << b << " : " << result << endl;

result = (a >= b);

cout << a << " >= " << b << " : " << result << endl;

result = (a <= b);

cout << a << " <= " << b << " : " << result << endl;

return 0;

}

**Problem 18:**

**Write a program that asks the user for three boolean values (1 for true, 0 for false). Implement**

**logical operations using logical operators (&&, ||, !). Display the results of each operation.**

**Code:**

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

int x, y, z;

bool result;

cout << "Enter three boolean values (0 or 1): ";

cin >> x >> y >> z;

result = (x && y);

cout << "(x && y) : " << result << endl;

result = (x || y);

cout << "(x || y) : " << result << endl;

result = !(x && y);

cout << "!(x && y) : " << result << endl;

result = (x && y) || z;

cout << "(x && y) || z : " << result << endl;

return 0;

}

**Problem 19:**

**Write a program that demonstrates the use of prefix and postfix increment and decrement**

**operators. Initialize two integers and show the values before and after applying each operator.**

**Code:**

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

int a = 5, b = 10;

int x, y;

cout << "Initial values: a = " << a << ", b = " << b << endl;

x = ++a;

cout << "Prefix increment: " << x << endl;

y = b++;

cout << "Postfix increment: " << y << endl;

cout << "Values after increment: a = " << a << ", b = " << b << endl;

x = --a;

cout << "Prefix decrement: " << x << endl;

y = b--;

cout << "Postfix decrement: " << y << endl;

cout << "Values after decrement: a = " << a << ", b = " << b << endl;

return 0;

}

**Problem 20:**

**Write a program that demonstrates the precedence and associativity of operators. Implement a**

**complex expression involving addition, subtraction, multiplication, and division. Display the**

**result step-by-step to show how the operators are evaluated.**

**Code:**

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

int a = 5, b = 10, c = 15;

cout << "a = " << a << ", b = " << b << ", c = " << c << endl;

cout << "a + b \* c = " << a + b \* c << endl;

cout << "(a + b) \* c = " << (a + b) \* c << endl;

cout << "a + b - c = " << a + b - c << endl;

cout << "a \* b / c = " << a \* b / c << endl;

return 0;

}

**Problem 21:**

**Write a program to calculate compound interest for a bank deposit. Prompt the user to enter**

**the principal amount, annual interest rate, and the number of years. Use arithmetic assignment**

**operators to update and display the amount at the end of each year, showing the compounded**

**interest. The formula for calculating compound interest annually using arithmetic assignment**

**operators is:**

**A(new) = A(old) ×(1+r/100)**

**Where:**

**● A(new) is the amount after adding interest for the current year.**

**● A(old) is the previous amount (initially the principal amount).**

**● r is the annual interest rate (as a percentage).**

**Code:**

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

double principal, rate, interest = 0;

int year;

cout << "Enter principal amount: ";

cin >> principal;

cout << "Enter annual interest rate (%): ";

cin >> rate;

cout << "Enter number of years: ";

cin >> year;

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

principal = principal \* (1 + rate / 100);

cout << "Year " << i << ": Amount = $" << fixed << setprecision(2) << principal << endl;

}

return 0;

}

**Problem 22:**

**Write a program that converts seconds into hours, minutes, and remaining seconds. Prompt the**

**user to enter a number of seconds. Use increment and decrement operators to calculate and**

**display the equivalent time in hours, minutes, and seconds.**

**Code:**

#include <iostream>

using namespace std;

int main() {

int sec, hour, min;

cout << "Enter number of seconds: ";

cin >> sec;

hour = sec / 3600;

sec = sec % 3600;

min = sec / 60;

sec = sec % 60;

cout << "hours = " << hour << ", minutes = " << min << ", sec = " << sec << endl;

return 0;

}

**Problem 23:**

**Write a program that takes four integer inputs from the user and determines the smallest**

**number among them.**

**Sample Output:**

**Enter four numbers: 3 5 2 8**

**The smallest number is: 2**

**Code:**

#include <iostream>

using namespace std;

int main() {

int a1, a2, a3, a4;

cout << "Enter four numbers: ";

cin >> a1 >> a2 >> a3 >> a4;

if (a1 < a2 && a1 < a3 && a1 < a4)

cout << a1 << " is the smallest number." << endl;

else if (a2 < a1 && a2 < a3 && a2 < a4)

cout << a2 << " is the smallest number." << endl;

else if (a3 < a1 && a3 < a2 && a3 < a4)

cout << a3 << " is the smallest number." << endl;

else

cout << a4 << " is the smallest number." << endl;

return 0;

}

**Problem 24:**

**Write a program that takes a character input from the user and checks if it is an alphabet (either**

**uppercase or lowercase).**

**Code:**

#include <iostream>

using namespace std;

int main() {

int b;

char a;

cout << "Enter a character: ";

cin >> a;

b = a;

if ((b >= 97 && b <= 122) || (b >= 65 && b <= 90))

cout << "The character is an alphabet.";

else

cout << "The character you entered is not an alphabet.";

return 0;

}

**Problem 25:**

**Write a program that takes a character input from the user and checks if it is a digit (0-9).**

**Code:**

#include <iostream>

using namespace std;

int main() {

int b;

char a;

cout << "Enter a character: ";

cin >> a;

b = a;

if (b >= 48 && b <= 57)

cout << "The character is a digit.";

else

cout << "The character you entered is not a digit.";

return 0;

}

**Problem 26:**

**Write a program that takes the purchase amount as input from the user and calculates the**

**discount based on the following criteria:**

**• amount >= 1000: 20% discount**

**• 500 <= amount < 1000: 10% discount**

**• amount < 500: No discount**

**After calculating the discount, display both the discount amount and the final amount after the**

**discount.**

**Code:**

#include <iostream>

using namespace std;

int main() {

float purchase, discount, discount\_amount, final\_amount;

cout << "Enter the purchase amount: ";

cin >> purchase;

if (purchase >= 1000)

discount = 0.2;

else if (purchase >= 500 && purchase < 1000)

discount = 0.1;

else

discount = 0;

discount\_amount = purchase \* discount;

final\_amount = purchase - discount\_amount;

cout << "The discount is: " << discount\_amount << endl;

cout << "The final amount after discount is: " << final\_amount;

return 0;

}

**Problem 27:**

**Write a program that takes the month number (1-12) as input from the user and prints the**

**corresponding season.**

**• 12, 1, 2: Winter**

**• 3, 4, 5: Spring**

**• 6, 7, 8: Summer**

**• 9, 10, 11: Autumn**

**Code:**

#include <iostream>

using namespace std;

int main() {

int num;

cout << "Enter a month number: ";

cin >> num;

if (num == 1 || num == 2 || num == 12)

cout << "The season is Winter";

else if (num == 3 || num == 4 || num == 5)

cout << "The season is Spring";

else if (num == 6 || num == 7 || num == 8)

cout << "The season is Summer";

else if (num == 9 || num == 10 || num == 11)

cout << "The season is Autumn";

else

cout << "Invalid number";

return 0;

}

**Problem 28:**

**Write a program that takes a character input from the user and checks if it is an uppercase or**

**lowercase letter.**

**Code:**

#include <iostream>

using namespace std;

int main() {

int b;

char a;

cout << "Enter a character: ";

cin >> a;

b = a;

if (b >= 97 && b <= 122)

cout << "The character is a small alphabet.";

else if (b >= 65 && b <= 90)

cout << "The character is a capital alphabet.";

else

cout << "The character you entered is not an alphabet.";

return 0;

}

**Problem 29:**

**Write a program that determines the ticket price based on the age of a person:**

**• Children (0-12): $5**

**• Teenagers (13-17): $7**

**• Adults (18-64): $10**

**• Seniors (65 and above): $6**

**Code:**

#include <iostream>

using namespace std;

int main() {

int age, ticket;

cout << "Enter your age: ";

cin >> age;

if (age >= 0 && age <= 12)

ticket = 5;

else if (age >= 13 && age <= 17)

ticket = 7;

else if (age >= 18 && age <= 64)

ticket = 10;

else

ticket = 6;

cout << "Your ticket price is: $" << ticket;

return 0;

}

**Problem 30:**

**Write a program that calculates the bonus for an employee based on their years of service:**

**• Less than 5 years: 10% of salary**

**• 5 to 10 years: 15% of salary**

**• More than 10 years: 20% of salary**

**Code:**

#include <iostream>

using namespace std;

int main() {

double salary, service, bonus;

cout << "Enter your salary: ";

cin >> salary;

cout << "Enter your years of experience: ";

cin >> service;

if (service >= 0 && service < 5)

bonus = 0.1;

else if (service >= 5 && service <= 10)

bonus = 0.15;

else

bonus = 0.2;

cout << "Your bonus is: $" << bonus \* salary;

return 0;

}

**Problem 31:**

**Write a program that assigns a grade based on a student's marks:**

**• Marks >= 90: A+**

**• Marks >= 80: A**

**• Marks >= 70: B**

**• Marks >= 60: C**

**• Marks >= 50: D**

**• Marks < 50: F**

**Code:**

#include <iostream>

using namespace std;

int main() {

int marks;

cout << "Enter your marks: ";

cin >> marks;

if (marks >= 90)

cout << "Your grade is A+";

else if (marks >= 80)

cout << "Your grade is A";

else if (marks >= 70)

cout << "Your grade is B";

else if (marks >= 60)

cout << "Your grade is C";

else if (marks >= 50)

cout << "Your grade is D";

else

cout << "Your grade is F";

return 0;

}

**Problem 32:**

**Write a C++ program that takes an integer input (1-12) representing a month and prints the month's name using a switch statement.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int num;

cout << "Enter the month number (1-12) :";

cin >> num;

switch (num)

{

case 1:

cout << "January";

break;

case 2:

cout << "February";

break;

case 3:

cout << "March";

break;

case 4:

cout << "April";

break;

case 5:

cout << "May";

break;

case 6:

cout << "June";

break;

case 7:

cout << "July";

break;

case 8:

cout << "August";

break;

case 9:

cout << "September";

break;

case 10:

cout << "October";

break;

case 11:

cout << "November";

break;

case 12:

cout << "December";

break;

default:

cout << "Invalid Number";

}

return 0;

}

**Problem 33:**

**The task is to create a C++ program that acts as a simple calculator based on user menuselection. The program should display a menu with options for addition, subtraction, multiplication, and division. Upon receiving user input for their choice, the program should prompt the user to enter two numbers. Depending on the selected operation, use a switch statement to perform the arithmetic operation on these numbers and display the result. Ensure the program handles division by zero gracefully.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int num;

int a, b;

cout << "1.Addition" << endl;

cout << "2.Subtraction" << endl;

cout << "3.Multiplication" << endl;

cout << "4.Division" << endl;

cout << "Enter your choice";

cin >> num;

cout << "Enter two numbers : ";

cin >> a >> b;

switch (num)

{

case 1:

cout << "Addition is " << a + b;

break;

case 2:

cout << "Subtraction is " << a - b;

break;

case 3:

cout << "Multiuplication is " << a \* b;

break;

case 4:

cout << "Division is " << (float)a / b;

break;

default:

cout << "Invalid number";

}

return 0;

}

**Problem 34:**

**Write a C++ program to print odd numbers from 1 to 10.**

**Code:**

**FOR LOOP :**

#include<iostream>

using namespace std;

int main()

{

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

{

cout << i << endl;

i++;

}

return 0;

}

**WHILE LOOP :**

#include<iostream>

using namespace std;

int main()

{

int i = 1;

while (i <= 10)

{

cout << i << endl;

i += 2;;

}

return 0;

}

**DO - WHILE LOOP :**

include<iostream>

using namespace std;

int main()

{

int i = 1;

do {

cout << i << endl;

i += 2;

} while (i <= 10);

return 0;

}

**Problem 35:**

**Write a C++ program to print even numbers from 3 to 15.**

**Code:**

**FOR LOOP :**

#include<iostream>

using namespace std;

int main()

{

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

{

cout << i << endl;

i++;

}

return 0;

}

**WHILE LOOP :**

#include<iostream>

using namespace std;

int main()

{

int i = 1;

while (i <= 10)

{

cout << i << endl;

i += 2;;

}

return 0;

}

**DO - WHILE LOOP :**

include<iostream>

using namespace std;

int main()

{

int i = 1;

do {

cout << i << endl;

i += 2;

} while (i <= 10);

return 0;

}

**Problem 36:**

**Write a C++ program to print numbers from 2 to 27 that are completely divisible by 3.**

**Code:**

**For loop :**

#include<iostream>

using namespace std;

int main()

{

for (int i = 2; i <= 27; i++)

{

if (i % 3 == 0)

cout << i << endl;

}

}

**WHILE LOOP** :

#include<iostream>

using namespace std;

int main()

{

int i = 2;

while (i <= 27)

{

if (i % 3 == 0)

cout << i << endl;

i++;

}

}

**DO - WHILE LOOP :**

#include<iostream>

using namespace std;

int main()

{

int i = 2;

do {

if (i % 3 == 0)

cout << i << endl;

i++;

} while (i <= 27);

}

**Problem 37:**

**Write a C++ program to print numbers from 2 to 23 that are completely divisible by 2 and 3.**

**Code:**

**DO - WHILE LOOP :**

#include<iostream>

using namespace std;

int main()

{

int i = 2;

do {

if (i % 2 == 0 && i % 3 == 0)

cout << i << endl;

i++;

} while (i <= 23);

}

**WHILE LOOP :**

#include<iostream>

using namespace std;

int main()

{

int i = 2;

while (i <= 23) {

if (i % 2 == 0 && i % 3 == 0)

cout << i << endl;

i++;

}

}

**FOR LOOP :**

#include<iostream>

using namespace std;

int main()

{

for (int i = 2; i <= 23; i++)

{

if (i % 2 == 0 && i % 3 == 0)

cout << i << endl;

}

}

**Problem 38:**

**Write a C++ program to print the table of 5.**

**Code:**

**For LOOP :**

#include<iostream>

using namespace std;

int main()

{

int num = 5;

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

{

cout << num << " x " << i << " = " << num \* i << endl;

}

}

**While loop:**

#include<iostream>

using namespace std;

int main()

{

int num = 5;

int i = 1;

while (i <= 10)

{

cout << num << " x " << i << " = " << num \* i << endl;

i++;

}

}

**DO - While LOOP :**

#include<iostream>

using namespace std;

int main()

{

int num = 5;

int i = 1;

do

{

cout << num << " x " << i << " = " << num \* i << endl;

i++;

} while (i <= 10);

}

**Problem 39:**

**Write a C++ program to print squares and cubes of numbers from 2 to 10.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

cout << "Number\tSquare\tCube" << endl;

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

{

cout << i << "\t" << i \* i << "\t" << i \* i \* i << endl;

}

}

**Problem 40:**

**Write a C++ program to print this series: 3, 6, 9, 12, 15, 18, 21. (Do not use the mod % operator).**

**Code:**

#include<iostream>

using namespace std;

int main()

{

for (int i = 3; i <= 21; i += 3)

{

cout << i << " ";

if (i == 21)

cout << "\b";

}

}

**Problem 41:**

**Write a C++ program to print this series: 1, 2, 4, 8, 16, 32, 64.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int num = 1;

cout << "1,";

for (int i = 1; i <= 6; i++)

{

num = num \* 2;

cout << num << ",";

}

cout << "\b";

}

**Problem 42:**

**Write a C++ program to print the first fifteen odd numbers in descending order.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

for (int i = 29; i >= 1; i--)

{

if (i % 2 != 0)

cout << i << endl;

}

}

**Problem 43:**

**Write a C++ program to input 10 numbers from the user and find out how many numbers are**

**even and odd.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int num, a = 0, b = 0;

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

{

cout << "Enter number :";

cin >> num;

if (num % 2 == 0)

a++;

else

b++;

}

cout << "Even Numbers: " << a << endl;

cout << "ODD Numbers: " << b;

}

**Problem 44:**

**Write a C++ program to input positive numbers from the user until the user enters zero and**

**display how many numbers are entered.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int num = 1;

int i = 0;

while (num != 0)

{

cout << "enter positive Number";

cin >> num;

if (num > 0)

i = i;

else if (num < 0)

{

cout << "Negative number" << endl;

i--;

}

i++;

}

cout << "Total Numbers :" << i - 1;

}

**Problem 45:**

**Write a C++ program to input a number from the user only if it's greater than the previous number, else display a message to the user to input a greater value than the previous value.**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int a, b;

cout << "Enter a number :";

cin >> a;

while (a < 0 || a == 0 || a>0)

{

b = a;

cout << "Enter a large number than previous number :";

cin >> a;

if (b >= a)

{

cout << "Not a larger number than previous one" << endl;

break;

}

}

}

**Problem 46:**

**Write a C++ program to print the square on the user screen.**

**Shape:**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Code:**

#include<iostream>

using namespace std;

int main()

{

for (int i = 1; i <= 55; i++)

{

cout << "\*";

if (i % 11 == 0)

cout << endl;

}

}

**Peoblem 47:**

**Write a program that generates a multiplication table for numbers from 1 to n. The value of n should be taken as input from the user.**

**Sample Output:**

**Enter the value of n: 5**

**Multiplication Table (1 to 5):**

**1 2 3 4 5**

**2 4 6 8 10**

**3 6 9 12 15**

**4 8 12 16 20**

**5 10 15 20 25**

**Code:**

#include<iostream>

using namespace std;

int main()

{

   int n,table;

   cout << "Enter the value of n: ";

   cin >> n;

   cout << "multiplication table (1 to 5):" << endl;

   for (int i = 1; i <= n; i++)

   {

         for (int j = 1; j <= 5; j++)

         {

               table = i \* j;

               cout << table << " ";

         }

         cout << endl;

   }

   return 0;

}

**Problem 48:**

**Write a program that prints Floyd's triangle with n rows. The value of n should be taken as input from the user.**

**Sample Output:**

**Enter the number of rows: 5**

**1**

**2 3**

**4 5 6**

**7 8 9 10**

**11 12 13 14 15**

**Code:**

#include<iostream>

using namespace std;

int main()

{

   int n,k=1;

   cout << "Enter the number of rows: ";

   cin >> n;

   for (int i = 1; i <= n; i++)

   {

         for (int j=1; j <= i; j++)

         {

           cout  <<k << " ";

               k++;

         }

         cout << endl;

   }

   return 0;

}

**Peoblem 49:**

**Write a program that calculates the grades of multiple students for multiple subjects. The user should input the number of students and the number of subjects. The program should ask for each student's scores for all subjects, calculate the average score, and determine the grade based on the average score using the following criteria:**

**• 90-100: A**

**• 80-89: B**

**• 70-79: C**

**• 60-69: D**

**• Below 60: F**

**Use a do-while loop to allow the user to repeat the process for a new set of students until they**

**decide to exit by entering a specific value.**

**Code:**

#include<iostream>

using namespace std;

char grade(double);

int main()

{

string ans; double avg, sum = 0.00;

int students, subject, j = 1, i=1,num,k;

do

{

cout << "Enter number of students: ";

cin >> students;

cout << "Enter the number of subjects: ";

cin >> subject;

cout << endl;

while (i<=students)

{

sum = 0;

k = 0;

cout << endl;

cout << "Enter scores for student " << i << endl;

for (int j = 1; j <= subject; j++)

{

cout << "Enter the number in subject " << j << " :";

cin >> num;

sum = sum + num;

k++;

}

avg = sum / k;

cout << endl;

cout << "Student " << i << ", Average marks " << avg<<",Grade = "<<grade(avg);

cout << endl;

i++;

}

cout << "Do you want to continue:(yes/no) ";

cin >> ans;

cout << endl;

i = 1;

} while (ans=="yes");

return 0;

}

char grade(double avg)

{

if (avg <= 100 && avg >= 90)

return 'A';

else if (avg >= 80 && avg <= 89)

return 'B';

else if (avg >= 70 && avg <= 79)

return 'C';

else if (avg >= 60 && avg <= 69)

return 'D';

else

return 'F';

}

**Problem 50:**

**Write a function isEven that takes an integer as an argument and returns true if the number is even, and false otherwise. Call this function from the main and print an appropriate message.**

**Code:**

#include<iostream>

using namespace std;

int isEven(int);

int main()

{

int num;

bool message;

cout << "Enter a number: ";

cin >> num;

message = isEven(num);

cout <<boolalpha<< message;

}

int isEven(int n)

{

bool a;

if (n % 2 == 0)

{

a=true;

}

else

a=false;

return a;

}

**Problem 51:**

**Write a function findMin that takes four integers as arguments and returns the minimum of the four. Call this function from main and print the result.**

**Code:**

#include<iostream>

using namespace std;

int findMin(int a, int b, int c, int d)

{

int min;

if ((a < b) && (a < c) && (a < d))

{

min = a;

}

else if ((b < a) && (b < c) && (b < d))

{

min = b;

}

else if ((c < a) && (c < b) && (c < d))

{

min = c;

}

else

{

min = d;

}

return min;

}

int main()

{

int a, b, c, d,y;

cout << "Enter four different numbers: ";

cin >> a >> b >> c >> d;

if ((a == b) || (b == c) || (c == d) || (a == c) || (a == d) || (b == d))

{

cout << "Numbers are not different." << endl;

return 0;

}

y = findMin(a, b, c, d);

cout << "Minimum number is " << y;

}

**Problem 52:**

**Write a function power that takes two integers, base and exponent, and returns the result of raising the base to the power of the exponent.**

**Code:**

#include<iostream>

using namespace std;

int power(int base, int exp)

{

int a = 1;

for (int i = 1; i <= exp; i++)

{

a=a\* base;

}

return a;

}

int main()

{

int base, exponent;

cout << "Enter the base: ";

cin >> base;

cout << "Enter the exponent: ";

cin >> exponent;

cout << "Answer of base to power is: " << power(base, exponent);

}

**Problem 53:**

**Write a C++ program that calculates the area of different shapes using function overloading.**

**Implement three different functions named area to calculate the area of:**

**1. A circle, which takes one argument (the radius) and returns a double representing the area using the formula: Area = π \* radius2.**

**2. A rectangle, which takes two arguments (the length and width) and returns a double representing the area using the formula: Area = length \* width.**

**3. A triangle, which takes two arguments (the base and height) and returns a double representing the area using the formula: Area = 0.5 \* base \* height.**

**Additionally, implement separate functions for taking inputs and displaying outputs for**

**each shape.**

**Code:**

#include<iostream>

using namespace std;

double area(float radius)

{

double area = 3.14\*(radius\*radius);

return area;

}

double area(float length, float width)

{

double area = length \* width;

return area;

}

double area(double length, double height)

{

double area = 0.5 \* length \* height;

return area;

}

int main()

{

float radius,x,length,width,z,y;

double base, height;

cout << "Enter the radius of circle: ";

cin >> radius;

x=area(radius);

cout << "Area of circle is: " << x<<endl;

cout << "Enter the length and width of rectangle: ";5

cin >> length >> width;

y = area(length, width);

cout << "Area of rectangle is: " << y<<endl;

cout << "Enter the base and height of triangle: ";

cin >> base >> height;

z = area(base, height);

cout << "Area of triangle is: " << z << endl;

}

**Problem 54:**

**Write a C++ program that demonstrates the concept of scope. Define a global variable named x and a local variable named x within a function. Implement a function named showScope that declares a local variable with the same name as the global variable and prints its value using the formula: Local x = 20. The main function should print the value of the global variable before and after calling the showScope function to show how the local variable shadows the global variable within the function. The function showScope should return void as it only prints the value of the local variable. Implement separate functions to display the global variable value.**

**Sample Output:**

**Global x: 10**

**Local x inside function: 20**

**Global x after function call: 10**

**Code:**

#include<iostream>

using namespace std;

int x = 10;

void showcase()

{

int x =20;

cout << "Local variable x inside function: " << x<<endl;

}

int main()

{

cout << "Global vaiable " << x<<endl;

showcase();

cout << "Global variable after function call: " << x;

}

**Problem 55:**

**Write a C++ program that includes a function to calculate the sum of all elements in a 1D array of integers.**

**Function Signature:**

**int sumArray(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

int sumarray(int arr[], int size)

{

int sum=0;

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

{

sum = sum + arr[i];

}

return sum;

}

int main()

{

int sum;

const int size=6;

int arr[size];

cout << "Enter the numbers in array";

inputarray(arr, size);

sum = sumarray(arr, size);

cout << "Sum of array is " << sum;

return 0;

}

**Problem 56:**

**Write a C++ program that includes a function to check whether a given 1D array of integers is sorted in non-decreasing order.**

**Function Signature:**

**bool isSorted(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

bool isSorted(int arr[], int size)

{

bool result = true;

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

{

if (arr[i] > arr[i + 1])

{

result = false;

break;

}

}

return result;

}

int main()

{

bool result;

const int size=6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

result = isSorted(arr, size);

if (result == true)

cout << "The array is sorted";

else

cout << "The array is not sorted";

return 0;

}

**Problem 57:**

**Write a C++ program that includes a function to find the most frequent element in a 1D array of integers. If there are multiple elements with the same highest frequency, print any one of them.**

**Function Signature:**

**int findMostFrequent(int arr[], int size);**

**Code:**

**Code**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

int findmostfrequent(int arr[], int size)

{

int frequency, last=0,value;

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

{

frequency = 0;

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

{

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

{

frequency++;

}

}

if (frequency > last)

{

last = frequency;

value = arr[i];

}

}

return value;

}

int main()

{

const int size = 6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

int result= findmostfrequent(arr, size);

cout << "Most frequent value is " << result;

}

**Problem 58:**

**Write a C++ program that includes a function to replace each element in a 1D array of integers with its square.**

**Function Signature:**

**void squareElements(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

void squareElements(int arr[], int size)

{

int square;

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

{

square = arr[i] \* arr[i];

arr[i] =square;

}

}

int main()

{

const int size = 6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

cout << "The array with square elements is : ";

squareElements(arr, size);

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

{

cout << arr[i] << " ";

}

}

**Problem 59:**

**Write a C++ program that includes a function to find all elements in a 1D array that are greater than the average of the array.**

**Function Signature:**

**void findElementsGreaterThanAverage(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

void findElementsGreaterThanAverage(int arr[], int size)

{

int sum = 0;

float average;

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

{

sum = sum + arr[i];

}

average = (float)sum / size;

cout << "The average of array is " << average<<endl;

cout << "Elements greater than average are : ";

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

{

if (arr[i] > average)

cout << arr[i] << " ";

}

}

int main()

{

const int size = 6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

findElementsGreaterThanAverage(arr, size);

}

**Problem 60:**

**Write a C++ program that includes a function to perform Bubble Sort on a 1D array of integers. The function should sort the array in ascending order.**

**Function Signature:**

**void bubbleSort(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

void bubblesort(int arr[], int size)

{

int temp;

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

{

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

{

if (arr[j] > arr[j + 1])

{

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

cout << "The sorted array is : ";

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

cout<< arr[i]<<" ";

}

}

int main()

{

const int size = 6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

bubblesort(arr, size);

}

**Problem 61:**

**Write a C++ program that includes a function to perform Selection Sort on a 1D array of integers. The function should sort the array in ascending order.**

**Function Signature:**

**void selectionSort(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

void selectionsort(int arr[], int size) {

int temp;

int minimum;

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

minimum = i;

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

if (arr[j] < arr[minimum]) {

minimum = j;

}

}

temp = arr[minimum];

arr[minimum] = arr[i];

arr[i] = temp;

}

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

cout<< arr[i]<<" ";

}

}

int main()

{

const int size = 6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

selectionsort(arr, size);

}

**Problem 62:**

**Write a C++ program that includes a function to perform Insertion Sort on a 1D array of integers. The function should sort the array in ascending order.**

**Function Signature:**

**void insertionSort(int arr[], int size);**

**Code:**

#include <iostream>

using namespace std;

void inputarray(int arr[], int size) {

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

cin >> arr[i];

}

}

void insertionsort(int arr[], int size) {

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

int key = arr[i];

int j = i - 1;

while (j >= 0 && arr[j] > key) {

arr[j + 1] = arr[j];

j--;

}

arr[j + 1] = key;

}

cout << "Sorted array is : ";

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

cout<< arr[i]<<" ";

}

}

int main()

{

const int size = 6;

int arr[size];

cout << "Enter the numbers in array : ";

inputarray(arr, size);

insertionsort(arr, size);

}

**Problem 63:**

**Write a C++ program that takes 7 values in arrays and displays it on the console.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[7];

    cout << "Enter 7 numbers: ";

    for (int i = 0; i < 7; i++) {

        cin >> num[i];

    }

    return 0;

}

**Problem 64:**

**Write a C++ program that initializes 10 values in arrays and displays them in reverse order on the console.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

    cout << "Original order: ";

    for (int i = 0; i < 10; i++) {

        cout << num[i] << " ";

    }

    cout << endl;

    cout << "Numbers in reverse order: ";

    for (int i = 9; i >= 0; i--) {

        cout << num[i] << " ";

    }

    return 0;

}

**Problem 65:**

**Write a C++ program that initializes 12 values in arrays and displays the sum of values at the 5th and 7th index of the array.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[12] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};

    int sum = num[5] + num[7];

    cout << "Sum of values at 5th and 7th index is " << sum;

    return 0;

}

**Problem 66:**

**Write a C++ program that declares an array of size 10 and initializes with values {2, 4, 6, 8, 10, ...,20} using the loop.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[10];

    int j = 2;

    for (int i = 0; i < 10; i++) {

        num[i] = j;

        cout << num[i] << " ";

        j = j + 2;

    }

    return 0;

}

**Problem 67:**

**Write a C++ program that takes 5 values in arrays and also takes a number n from the user. Print all values of the array that are greater than n.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[5];

    int n;

    cout << "Enter 5 values: ";

    for (int i = 0; i < 5; i++) {

        cin >> num[i];

    }

    cout << "Enter the number from which you want greater numbers: ";

    cin >> n;

    cout << "Numbers greater than " << n << " are: ";

    for (int i = 0; i < 5; i++) {

        if (num[i] > n)

            cout << num[i] << " ";

    }

    return 0;

}

**Problem 68:**

**Write a C++ program that initialize 16 values in arrays and print only even index values.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[16] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16};

    cout << "Even index values are: ";

    for (int i = 0; i < 16; i++) {

        if (i % 2 == 0)

            cout << num[i] << " ";

    }

    return 0;

}

**Problem 69:**

**Write a program that initializes 16 values in arrays and prints only even values (Note: This is different from the above task).**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[16] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16};

    cout << "Even values are: ";

    for (int i = 0; i < 16; i++) {

        if (num[i] % 2 == 0)

            cout << num[i] << " ";

    }

    return 0;

}

**Problem 70:**

**Write a C++ program that initializes 10 values in arrays and counts the number of even and odd values in arrays.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

    int a = 0, b = 0;

    cout << "Values of array are: ";

    for (int i = 0; i < 10; i++) {

        cout << num[i] << " ";

    }

    cout << endl;

    for (int i = 0; i < 10; i++) {

        if (num[i] % 2 == 0)

            a++;

        else

            b++;

    }

    cout << "Even values count: " << a << endl;

    cout << "Odd values count: " << b;

    return 0;

}

**Problem 71:**

**Write a C++ program that initializes 10 values in arrays. Your program asks for a number from the user and finds whether the entered number is present in the array or not.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

    int a, b = 0;

    cout << "Values of array are: ";

    for (int i = 0; i < 10; i++) {

        cout << num[i] << " ";

    }

    cout << endl;

    cout << "Enter a number you want to check whether it is present in the array or not: ";

    cin >> a;

    for (int j = 0; j < 10; j++) {

        if (a == num[j]) {

            b = 1;

            cout << "Present";

            break;

        }

    }

    if (b != 1) {

        cout << "Not present";

    }

    return 0;

}

**Problem 72:**

**Write a C++ program that initializes 10 values in arrays. Your program asks for a number from the user and finds whether the entered number is present in the array or not.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[10] = {1, 1, 3, 4, 5, 6, 7, 8, 9, 10};

    int a, frequency = 0;

    cout << "Values of array are: ";

    for (int i = 0; i < 10; i++) {

        cout << num[i] << " ";

    }

    cout << endl;

    cout << "Enter a number you want to check its frequency: ";

    cin >> a;

    for (int j = 0; j < 10; j++) {

        if (num[j] == a)

            frequency++;

    }

    cout << "The number repeats " << frequency << " times in the array.";

    return 0;

}

**Problem 73:**

**Write a C++ program that initializes 10 values in arrays. Your program asks for a number N from the user and replaces it with another number T entered by the user.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

    int n, t, b = 0, j;

    cout << "Values of array are: ";

    for (int i = 0; i < 10; i++) {

        cout << num[i] << " ";

    }

    cout << endl;

    cout << "Enter a number you want to check whether it is present in the array or not: ";

    cin >> n;

    for (j = 0; j < 10; j++) {

        if (n == num[j]) {

            b = 1;

            cout << "Present";

            break;

        }

    }

    if (b != 1) {

        cout << "Not present";

        return 0;

    }

    cout << endl;

    cout << "Enter the number by which you want to replace it: ";

    cin >> t;

    num[j] = t;

    cout << "New array is: ";

    for (int i = 0; i < 10; i++) {

        cout << num[i] << " ";

    }

    return 0;

}

**Problem 74:**

**Write a C++ program that merges two arrays into a single array and displays the resulting array.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num1[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

    int num2[5] = {11, 12, 13, 14, 15};

    int num3[15];

    cout << "Total elements in first array: 10" << endl;

    cout << "Values of 1st array: ";

    for (int i = 0; i < 10; i++) {

        cout << num1[i] << " ";

    }

    cout << endl;

    cout << "Total elements in second array: 5" << endl;

    cout << "Values of 2nd array: ";

    for (int i = 0; i < 5; i++) {

        cout << num2[i] << " ";

    }

    cout << endl;

    // Copy first array into num3

    for (int i = 0; i < 10; i++) {

        num3[i] = num1[i];

    }

    // Copy second array into num3

    int j = 0;

    for (int i = 10; i < 15; i++) {

        num3[i] = num2[j];

        j++;

    }

    // Display merged array

    cout << "Merged array: ";

    for (int i = 0; i < 15; i++) {

        cout << num3[i] << " ";

    }

    return 0;

}

**Problem 75:**

**Write a program that prompts the user to enter two strings: one string containing spaces and another string without any spaces. The program should then display both strings.**

**Code:**

#include <iostream>

using namespace std;

void displayStrings(const char strWithSpaces[], const char strWithoutSpaces[])

{

cout << "String with spaces is " << strWithSpaces<<endl;

cout << "String without spaces is " << strWithoutSpaces << endl;

}

int main()

{

char withspaces[100];

char withoutspaces[100];

cout << "Enter a string with spaces: ";

cin.get(withspaces, 100);

cin.ignore();

cout << "Enter a string without spaces: ";

cin >> withoutspaces;

displayStrings(withspaces, withoutspaces);

return 0;

}

**Problem 76:**

**Write a function that returns the length of a string stored in a one-**

**dimensional char array without using strlen.**

**Code:**

#include <iostream>

using namespace std;

int stringLength(const char str[])

{

//cout<<"The length of a string is :"<<str[]

for (int i = 0; i < 100; i++)

{

if (str[i] == '\0')

{

cout << "The length of a string is : " << i;

break;

}

}

return 0;

}

int main()

{

char str[100];

cout << "Enter a string: ";

cin.get(str, 100);

stringLength(str);

return 0;

}

**Problem 77:**

**Write a function that takes a one-dimensional char array as input and**

**reverses the string.**

**Code:**

#include <iostream>

using namespace std;

void swap(char str1[], int size1, int size2)

{

char temp;

temp = str1[size1];

str1[size1] = str1[size2];

str1[size2] = temp;

}

void reverseString(char str[])

{

int count = -1;

int start(0);

int i = 0;

while (str[i++] != '\0')

{

count++;

}

int end = count;

while (start < end)

{

swap(str, start, end);

start++;

end--;

}

cout << "Reverse String is: " << str;

}

int main()

{

char str[100];

cout << "Enter a string: ";

cin.get(str, 100);

//cout << str;

reverseString(str);

return 0;

}

**Problem 78:**

**Write a function that concatenates two one-dimensional char arrays**

**into one.**

**Code:**

#include <iostream>

using namespace std;

void concatenateStrings(char dest[], const char src[])

{

int count1 = 0;

while (dest[count1] != '\0')

{

count1++;

}

int count2 = 0;

while (src[count2] != '\0')

{

count2++;

}

for (int a = 0; a < count2; a++)

{

dest[count1 + a] = src[a];

}

dest[count1 + count2] = '\0';

cout << "Concatenate array is : " << dest;

}

int main()

{

char str1[100], str2[100];

cout << "Enter the string 1 : ";

cin.get(str1, 100);

cin.ignore();

cout << "Enter the string 2 : ";

cin.get(str2, 100);

concatenateStrings(str1, str2);

return 0;

}

**Problem 79:**

**Write a function that converts all the characters in a one-dimensional char array to uppercase.**

**Code:**

#include <iostream>

using namespace std;

void toUpperCase(char str[])

{

int i = 0;

while (str[i] != '\0')

{

if(str[i]>='a'&&str[i]<='z')

str[i] = str[i] - 32;

i++;

}

cout << str;

}

int main()

{

char str1[100];

cout << "Enter the string : ";

cin.get(str1, 100);

toUpperCase(str1);

return 0;

}

**Problem 80:**

**Write a function that counts the number of vowels and consonants in a**

**one-dimensional char array.**

**Code:**

#include <iostream>

using namespace std;

void countVowelsConsonants(const char str[], int& vowels, int& consonants)

{

int i = 0;

while (str[i] != '\0')

{

char a=str[i];

if (a == 'a'||a=='e'||a=='i'||a=='o'||a=='u')

{

vowels++;

}

else

consonants++;

i++;

}

}

int main()

{

int vowel = 0,cons=0;

char str[100];

cout << "Enter a string: ";

cin.get(str, 100);

countVowelsConsonants(str, vowel, cons);

cout << "Vowels: " << vowel << "\t Consonents: " << cons;

return 0;

}

**Problem 81:**

**Write a function that replaces all occurrences of a specific character in a one-dimensional char array with another character.**

**Code:**

#include <iostream>

using namespace std;

void replaceCharacter(char str[], char &oldChar, char &newChar)

{

int count = 0;

while (str[count] != '\0')

{

count++;

}

for (int i = 0; i < count; i++)

{

if (str[i] == oldChar)

{

str[i] = newChar;

}

}

}

int main()

{

char old,current;

char str[100];

cout << "Enter a string: ";

cin.get(str, 100);

cout << "Enter the charcter from string which you want to replace: ";

cin >> old;

cout << "Enter the charcter by which you want to replace: ";

cin >> current;

replaceCharacter(str, old,current);

cout << "New string is : " << str;

return 0;

}

**Problem 82:**

**8: Write a function that removes all duplicate characters from a one-**

**dimensional char array.**

**Code:**

#include <iostream>

using namespace std;

void removeDuplicates(char str[])

{

int count = 0;

while (str[count] != '\0')

{

count++;

}

for (int i = 0; i < count; i++)

{

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

{

if (str[i] == str[j])

{

str[j] = '1';

}

}

if (str[i] != '1')

{

cout << str[i];

}

// for (int i = 0; i < count; i++)

/// {

// if (str[i] == '1')

// {

// str[i] = '\0';

// }

// }

// cout << endl<<str;

}

}

int main()

{

char ary[30];

cout << "Enter the string to reverse it: ";

cin >> ary;

removeDuplicates(ary);

return 0;

}

**Problem 83:**

**Write a program that dynamically allocates a char array based on the**

**user's input size. Implement these functions:**

**Code:**

#include <iostream>

using namespace std;

void inputCharArray(char ary[], int size)

{

cout << "Enter the characters in array: ";

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

{

cin >> ary[i];

}

}

void displayCharArray(char ary[],int size)

{

cout << "The entered character in array are: ";

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

{

cout << ary[i];

}

}

int main()

{

int size;

cout << "Enter the size of array: ";

cin >> size;

char\* ary = new char[size];

inputCharArray(ary, size);

displayCharArray(ary,size);

delete[] ary;

return 0;

}

**Problem 84:**

**10: Write a program where the char array is used to store a user's string input. Implement these functions:**

** allocateArray: Allocates memory for the array.**

** inputCharArray: Reads the string from the user.**

** replaceSpacesWithUnderscores: Replaces all spaces in the array with**

**underscores.**

** displayCharArray: Displays the modified string.**

** deallocateArray: Frees the allocated memory.**

**Code:**

**Problem 85:**

**Code:**

**Problem 86:**

**Code:**

#include <iostream>

using namespace std;

char \*allocateArray()

{

char \*str = new char[20];

return str;

}

void inputCharArray(char str[])

{

cout << "Enter the string: ";

cin.get(str, 20);

}

void replaceSpacesWithUnderscores(char str[])

{

for (int i = 0; i < 20; i++)

{

if (str[i] == ' ')

{

str[i] = '\_';

}

}

}

void displayCharArray(char str[])

{

cout << "New string is: "<<str;

}

void deallocateArray(char \*str)

{

delete[] str;

}

int main()

{

char\* str= allocateArray();

inputCharArray(str);

replaceSpacesWithUnderscores(str);

displayCharArray(str);

deallocateArray(str);

return 0;

}

**Problem 87:**

**The task is to develop a simple program to manage student records for a small classroom. Each**

**student has a name, roll number, and marks in three subjects (Math, Science, and English).**

**● Define a struct named Student with members for storing the student's name, roll**

**number, and marks in three subjects.**

**● Create three individual Student variables.**

**● Write a program that inputs the details for each student and then displays the details**

**along with the total marks for each student.**

**Code:**

#include<iostream>

#include<string>

using namespace std;

struct student {

string name;

int roll = 1;

int marks1 = 1;

int marks2 = 1;

int marks3 = 1;

//int total = marks1 + marks2 + marks3;

};

int main()

{

student p1, p2, p3;

int a1, a2, a3;

cout << "Enter the name of Student 1: ";

getline(cin, p1.name);

// cin.ignore();

cout << "Enter the roll# of student 1: ";

cin >> p1.roll;

cout << "Enter the marks of Student 1 in math: ";

cin >> p1.marks1;

cout << "Enter the marks of Student 1 in science: ";

cin >> p1.marks2;

cout << "Enter the marks of Student 1 in english: ";

cin >> p1.marks3;

a1 = p1.marks1 + p1.marks2 + p1.marks3;

cin.ignore();

cout << "Enter the name of Student 2: ";

getline(cin, p2.name);

//cin.ignore();

cout << "Enter the roll# of student 2: ";

cin >> p2.roll;

cout << "Enter the marks of Student 2 in math: ";

cin >> p2.marks1;

cout << "Enter the marks of Student 2 in science: ";

cin >> p2.marks2;

cout << "Enter the marks of Student 2 in english: ";

cin >> p2.marks3;

a2 = p2.marks1 + p2.marks2 + p2.marks3;

cin.ignore();

cout << "Enter the name of Student 3: ";

getline(cin, p3.name);

//cin.ignore();

cout << "Enter the roll# of student 3: ";

cin >> p3.roll;

cout << "Enter the marks of Student 3 in math: ";

cin >> p3.marks1;

cout << "Enter the marks of Student 3 in science: ";

cin >> p3.marks2;

cout << "Enter the marks of Student 3 in english: ";

cin >> p3.marks3;

a3 = p3.marks1 + p3.marks2 + p3.marks3;

cout << "Student 1:" << p1.name << " ,Roll number: " << p1.roll << ", Total marks: " << a1 << endl;

cout << "Student 2:" << p2.name << " ,Roll number: " << p2.roll << ", Total marks: " << a2 << endl;

cout << "Student 3:" << p3.name << " ,Roll number: " << p3.roll << ", Total marks: " << a3 << endl;

return 0;

}

**Problem 88:**

**The task is to develop a simple program to keep track of cars in a garage. Each car has a model**

**name, year of manufacture, and mileage.**

**● Define a struct named Car with members for storing the car's model name, year of**

**manufacture, and mileage.**

**● Create one Car variable.**

**● Write a program that inputs the details for the car and then displays the details.**

**Code:**

#include<iostream>

#include<string>

using namespace std;

struct car {

int year;

string model;

int avg;

};

int main()

{

car p1;

cout << "Enter the Details of car: " << endl;

cout << "Model Name: ";

getline(cin, p1.model);

cout << "Manufacture: ";

cin >> p1.year;

cout << "Mileage: ";

cin >> p1.avg;

cout << endl;

cout << "Car: " << p1.model << ", Year: " << p1.year << ", Mileage: " << p1.avg;

}

**Problem 89:**

**The task is to create a program to manage books in a library. Each book has a title, author, and**

**ISBN number.**

**● Define a struct named Book with members for storing the book's title, author, and**

**ISBN number.**

**● Create two individual Book variables.**

**● Write a program that inputs the details for each book and then displays the details.**

**Code:**

#include<iostream>

#include<string>

using namespace std;

struct book {

string book;

string author;

string isbn;

};

int main()

{

book b1, b2;

cout << "Enter details for book 1"<<endl;

cout << "Title: ";

getline(cin,b1.book);

cout << "Author: ";

getline(cin, b1.author);

cout << "ISBN: ";

getline(cin, b1.isbn);

cout << "Enter details for book 2" << endl;

cout << "Title: ";

getline(cin, b2.book);

cout << "Author: ";

getline(cin, b2.author);

cout << "ISBN: ";

getline(cin, b2.isbn);

cout << endl;

cout << "Book 1: " << "\"" << b1.book << "\"" << " by " << b1.author << ", ISBN: " << b1.isbn<<endl;

cout << "Book 2: " << "\"" << b2.book << "\"" << " by " << b2.author << ", ISBN: " << b2.isbn << endl;

return 0;

}

**Problem 90:**

**The task is to develop a program to manage employee salaries in a company. Each employee**

**has a name, ID, basic salary, and a bonus percentage.**

**● Define a struct named Employee with members for storing the employee's name, ID,**

**basic salary, and bonus percentage.**

**● Create two individual Employee variables.**

**● Write a program that inputs the details for each employee and then calculates and**

**displays their total salary, which is the sum of the basic salary and the bonus amount.**

**Code:**

#include<iostream>

#include<string>

using namespace std;

struct employee {

string name;

int id;

float basic;

float bonus;

};

int main()

{

float sal1,sal2;

employee e1, e2;

cout << "Enter the deatails of employee 1:" << endl;

cout << "Name: ";

getline(cin, e1.name);

cout << "ID: ";

cin >> e1.id;

cout << "Basic Salary: ";

cin >> e1.basic;

cout << "Bonus in %: ";

cin >> e1.bonus;

cout << endl;

e1.bonus = e1.basic \* (e1.bonus / 100);

sal1 = e1.bonus + e1.basic;

cin.ignore();

cout << "Enter the deatails of employee 2:" << endl;

cout << "Name: ";

getline(cin, e2.name);

cout << "ID: ";

cin >> e2.id;

cout << "Basic Salary: ";

cin >> e2.basic;

cout << "Bonus in %: ";

cin >> e2.bonus;

e2.bonus = e2.basic \* (e2.bonus / 100);

sal2 = e2.bonus + e2.basic;

cout << endl;

cout << "Employee 1: " << e1.name << ", ID: " << e1.id << ", Total Salary: " << sal1 << endl;

cout << "Employee 2: " << e2.name << ", ID: " << e2.id << ", Total Salary: " << sal2 << endl;

return 0;

}

**Problem 91:**

**Write a C++ program that:**

**• Creates a function inputArray to read n integers into an array.**

**• Passes the array to a function reverseArray that reverses the elements.**

**• Uses a function outputArray to display the reversed array.**

**Code:**

#include<iostream>

using namespace std;

void inputarray(int arr[], int size)

{

cout << "Please enter the numbers in the array: ";

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

{

cin >> arr[i];

}

}

void reversearray(int arr[], int size)

{

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

{

cout << arr[i] << " ";

}

}

void outputarray(int arr[], int size)

{

cout << "Reverse array is :";

cout << endl;

reversearray(arr, size);

}

int main()

{

int size;

cout << "Enter the size of array: ";

cin >> size;

int\*arr=new int [size];

inputarray(arr, size);

outputarray(arr, size);

delete [] arr;

return 0;

}

**Problem 92:**

**Write a C++ program that:**

**• Creates a function inputArray to read n integers into an array.**

**• Passes the array to a function countEvenOdd that counts the number of even and odd numbers.**

**• Uses a function outputCounts to display the count of even and odd numbers.**

**Code:**

#include<iostream>

using namespace std;

void inputarray(int arr[], int size)

{

cout << "Please enter the numbers in the array: ";

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

{

cin >> arr[i];

}

}

void countevenodd(int arr[], int size, int& evencount, int& oddcount)

{

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

{

if (arr[i] % 2 == 0)

{

evencount++;

}

else

{

oddcount++;

}

}

}

void outputcounts(int counteven, int countodd)

{

cout << "Total even numbers in array are " << counteven << endl;

cout << "Total odd numbers in array are " << countodd;

}

int main()

{

int even = 0, odd = 0;

int size;

cout << "Enter the size of array: ";

cin >> size;

int \*arr=new int[size];

inputarray(arr, size);

countevenodd(arr, size, even, odd);

outputcounts(even, odd);

delete [] arr;

return 0;

}

**Problem 93:**

**Write a C++ program that:**

**• Creates a function inputArray to read n integers into an array.**

**• Passes the array to a function findFrequency that calculates the frequency of each element.**

**• Uses a function outputFrequencies to display each element and its frequency.**

**Code:**

Code

#include<iostream>

using namespace std;

void inputarray(int arr[], int size)

{

cout << "Please enter the numbers in the array: ";

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

{

cin >> arr[i];

}

}

void findFrequency(int arr[], int size, int freq[])

{

int frequency;

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

{

if (arr[i] == -1)

{

continue;

}

frequency = 1;

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

{

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

{

frequency++;

arr[j] = -1;

}

}

freq[i] = frequency;

}

}

void outputfrequencies(int arr[], int size, int freq[])

{

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

{

if (arr[i] != -1)

{

cout << arr[i] << " is repeated " << freq[i] << " times." << endl;

}

}

}

int main()

{

int size;

cout << "Enter the size of array: ";

cin >> size;

int\* arr=new int[size];

int \*freq=new int[size];

inputarray(arr, size);

findFrequency(arr, size, freq);

outputfrequencies(arr, size, freq);

delete [] arr;

delete[] freq;

}

**Problem 94:**

**Write a C++ program that:**

**• Creates two functions inputArray1 and inputArray2 to read n integers into two separate arrays.**

**• Creates a function mergeArrays that merges the two arrays into a third array.**

**• Creates a function sortArray that sorts the merged array.**

**• Creates a function outputArray to display the sorted, merged array.**

**Code:**

#include<iostream>

using namespace std;

void inputarray1(int arr1[], int size1)

{

for (int i = 0; i < size1; i++)

{

cin >> arr1[i];

}

}

void inputarray2(int arr2[], int size2)

{

for (int i = 0; i < size2; i++)

{

cin >> arr2[i];

}

}

void mergearrays(int arr1[], int size1, int arr2[], int size2, int mergedarray[])

{

for (int i = 0; i < size1; i++)

{

mergedarray[i] = arr1[i];

}

for (int i = 0; i < size2; i++)

{

mergedarray[i + size1] = arr2[i];

}

}

void sortarray(int arr[], int size)

{

int temp;

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

{

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

{

if (arr[j] > arr[j + 1])

{

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

void outputArray(int arr[], int size)

{

cout << "Merged and sorted array is : " << endl;

sortarray(arr, size);

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

{

cout << arr[i] << " ";

}

}

int main()

{

int size1;

int\* arr1=new int[size1];

int size2 ;

int \*arr2=new int[size2];

cout << "Enter the size of array1 :";

cin >> size1;

int \*mergedarr=new int[size1+size2];

cout << "Enter the numbers in array 1: " << endl;

inputarray1(arr1, size1);

cout << "Enter the size of array2 :";

cin >> size2;

cout << "Enter the numbers in array 2: " << endl;

inputarray2(arr2, size2);

mergearrays(arr1, size1, arr2, size2, mergedarr);

outputArray(mergedarr, size1 + size2);

delete [] arr1;

delete[]arr2;

delete[] mergedarr;

return 0;

}

**Problem 95:**

**Write a C++ program that:**

**• Creates a function inputMatrix to read integers into a 3x4 matrix.**

**• Pass the matrix to a function rowSums to calculate and store the sum of each row into a 1D array.**

**• Use the function outputSums to display the sum of each row.**

**Code:**

#include <iostream>

using namespace std;

void inputmatrix(int mat[3][4])

{

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

{

for (int j = 0; j < 4; j++)

{

cin >> mat[i][j];

}

}

}

void rowsum(int mat[3][4],int arr[],int size)

{

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

{

int sum = 0;

for (int j = 0; j < 4; j++)

{

sum = sum + mat[i][j];

arr[i] = sum;

}

}

}

void outputsums(int arr[],int size)

{

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

{

cout << "Sum of row " << i << " is " << arr[i]<<endl;

}

}

int main()

{

int mat[3][4];

int arr[3];

cout << "Enter the numbers in array of 3 by 4 ";

inputmatrix(mat);

rowsum(mat, arr, 3);

outputsums(arr, 3);

}

**Problem 96:**

**Write a program to manage student data for a class of 20 students. The program will:**

**1. Read the names of the students and their test scores.**

**2. Assign grades to each student based on their test scores.**

**3. Identify and print the highest test score along with the names of the students who achieved that score.**

**Code:**

#include <iostream>

using namespace std;

struct studentType

{

string first\_name="none";

string last\_name="none";

int test\_score=0;

char grade='N';

};

void readStudentData(studentType arr[],const int size)

{

cout << "Enter Student Data:" << endl;

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

{

cout << "Student " << i + 1 << ":" << endl;

cout << "Enter the first name: ";

cin >> arr[i].first\_name;

cout << "Enter the last name: ";

cin >> arr[i].last\_name;

cout << "Enter the marks obtained: ";

cin >> arr[i].test\_score;

}

}

void assignGrades(studentType arr[],int size)

{

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

{

if (arr[i].test\_score >= 90)

arr[i].grade = 'A';

else if (arr[i].test\_score >= 80)

arr[i].grade = 'B';

else if (arr[i].test\_score >= 70)

arr[i].grade = 'C';

else if (arr[i].test\_score >= 60)

arr[i].grade = 'D';

else

arr[i].grade = 'F';

}

}

void displaydata(studentType arr[], int size)

{

cout << endl<<"Student Data:" << endl;

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

{

cout << arr[i].first\_name << " " << arr[i].last\_name << " : Test Score: " << arr[i].test\_score << ", Grade: "

<< arr[i].grade << endl;

}

}

int findHighestScore(studentType arr[], int size)

{

int highest = arr[0].test\_score;

int current;

for (int i = 1; i < size; i++)

{

current = arr[i].test\_score;

if (current > highest)

highest = current;

}

return highest;

}

void printStudentsWithHighestScore(studentType arr[], int size,int highest)

{

cout << "Student with highest Test Score are: " << endl;

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

{

if (arr[i].test\_score == highest)

cout << arr[i].first\_name << " " << arr[i].last\_name << endl;

}

}

int main()

{

int highest;

const int size = 20;

studentType arr[size];

readStudentData(arr, size);

assignGrades(arr,size);

displaydata(arr, size);

highest=findHighestScore(arr, size);

cout << endl<<endl;

cout << "Highest Test Score: " << highest << endl << endl;

printStudentsWithHighestScore(arr, size, highest);

return 0;

}

**Problem 97:**

**Write a C++ program that processes a sentence from a file named dataFile.txt. The program should first read the content of the file, then remove any punctuation marks such as commas, periods, and semicolons. After cleaning the sentence, the program should split it into individual words and calculate the length of each word. Finally, the program should write the words along with their respective lengths to a new file, output.txt, in the format word length, where each word and its length appear on a separate line. This task should be accomplished using functions to ensure modularity and clarity in the code.**

**Code:**

#include <iostream>

#include<string>

#include<fstream>

using namespace std;

void checkfile(ifstream &Inputfile,ofstream &Outputfile)

{

if (!Inputfile)

{

cout << "Error";

return ;

}

if (!Outputfile)

{

cout << "error";

return ;

}

}

string checkpunctuation(string line)

{

string cleanline;

for (char ch : line)

{

if (ch != ',' && ch != '.' && ch != ';' && ch != ':' && ch != '?' && ch != '!' &&

ch != '"' && ch != '\'' && ch != '(' && ch != ')' && ch != '-' && ch != '\_' &&

ch != '[' && ch != ']' && ch != '{' && ch != '}')

{

cleanline = cleanline + ch;

}

}

return cleanline;

}

void function(ofstream &Outputfile,string line)

{

string word;

for (char ch : line)

{

if (ch == ' ')

{

if (!word.empty())

{

Outputfile << word << " " << word.length() << endl;

word.clear();

}

}

else

word = word + ch;

}

if (!word.empty()) {

Outputfile << word << " " << word.length() << endl;

}

}

int main()

{

string line;

ifstream Inputfile("data.txt");

ofstream Outputfile("output.txt");

checkfile(Inputfile,Outputfile);

while (getline(Inputfile, line))

{

line = checkpunctuation(line);

function(Outputfile, line);

}

Inputfile.close();

Outputfile.close();

}

**Problem 98:**

**ActiveLife Fitness Center is a community gym that offers various membership plans to cater to the fitness needs of its clients. To ensure accurate and transparent billing, ActiveLife has decided to implement a program to calculate membership fees based on the type of membership and the number of months the client wishes to enroll.**

**The membership fee structure is as follows:**

**1. Basic Membership:**

**o $30 per month for a maximum of 6 months.**

**o If the membership is extended beyond 6 months, the rate is reduced to $25 per month.**

**2. Premium Membership:**

**o $50 per month for any duration.**

**o However, if the client signs up for more than 12 months, the monthly fee is reduced to $45.**

**3. Family Membership:**

**o A flat rate of $80 per month for up to 4 members.**

**o For each additional member, the rate increases by $10 per month.**

**As a programmer for ActiveLife, you need to develop a program that achieves the following:**

**• Prompt the user to select the type of membership (1 for Basic, 2 for Premium, 3 for Family).**

**• Ask the user to enter the number of months for which they want to enroll and the number of family members (if applicable).**

**• Use a switch statement to calculate the total membership fee based on the selected membership type.**

**• Display the final membership fee.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int choice, month, member;

    double fee;

    cout << "Welcome to ActiveLife Fitness Center!" << endl;

    cout << "Please select the type of membership:" << endl;

    cout << "1. Basic\n2. Premium\n3. Family" << endl;

    cout << "Enter your membership type (1-3): ";

    cin >> choice;

    if (choice <= 0 || choice > 3) {

        cout << "Invalid choice selected" << endl;

        return 0;

    }

    cout << "Enter the number of months: ";

    cin >> month;

    if (month <= 0) {

        cout << "Invalid month number." << endl;

        return 0;

    }

    switch (choice) {

        case 1:

            if (month <= 6) {

                fee = 30 \* month;

            } else {

                fee = 25 \* month;

            }

            break;

        case 2:

            if (month <= 12) {

                fee = 50 \* month;

            } else {

                fee = 45 \* month;

            }

            break;

        case 3:

            cout << "Enter the number of family members: ";

            cin >> member;

            if (member <= 4) {

                fee = 80 \* month;

            } else {

                fee = (80 \* month) + (10 \* (member - 4) \* month);

            }

            break;

    }

    cout << "Your total membership fee is: $" << fee << endl;

    return 0;

}

**Problem 99:**

**Number Crunchers Academy has introduced a new program aimed at teaching students about special numbers in mathematics. As part of this initiative, students are tasked with writing a program to verify whether a given number is an Armstrong number.**

**An Armstrong number for a three-digit number is defined as a number where the sum of its digits raised to the power of three equals the number itself. For example, 370 is an Armstrong number:**

**The program will ask users to input a three-digit number and check if it is an Armstrong number.**

**Code:**

#include <iostream>

using namespace std;

int main() {

    int num, a, b, c, q;

    cout << "Enter a three-digit number: ";

    cin >> num;

    q = num;

    a = num % 10;

    num = num / 10;

    b = num % 10;

    num = num / 10;

    c = num % 10;

    if ((a \* a \* a) + (b \* b \* b) + (c \* c \* c) == q) {

        cout << q << " is an Armstrong number." << endl;

    } else {

        cout << q << " is not an Armstrong number." << endl;

    }

    return 0;

}

**Problem 100:**

**Write a C++ program that allows a user to enter details of multiple warriors, save their information in a file, retrieve the data from the file, and simulate a battle between the first two warriors. Ensure that attack power is always greater than defense power and display the battle sequence with health updates. The program should also save the battle log in the file."**

**Code:**

#include<iostream>

#include<string>

#include<fstream>

using namespace std;

struct DATA {

    string name;

    int health;

    int attack;

    int defense;

};

int main() {

    int num;

    cout << "Name: Najmul Arifeen\nRoll #: fa24-bse-102\n"; // Name and roll #

    cout << "Enter the number of warriors: ";

    cin >> num;

    DATA\* data = new DATA[num]; // dynamically creating array

    ofstream file("data.txt");

    if (!file) {

        cout << "Error: Unable to create file." << endl; // Fixed error message

        return 1;

    }

    for (int i = 0; i < num; i++) {

        cout << "Enter details for Warrior " << i + 1 << ":\n";

        cout << "Name: ";

        cin >> data[i].name;

        cout << "Health: ";

        cin >> data[i].health; // details of warriors

        do {

            cout << "Attack power must be greater than defense power.\n";

            cout << "Attack Power: ";

            cin >> data[i].attack;

            cout << "Defense Power: ";

            cin >> data[i].defense;

        } while (data[i].attack < data[i].defense);

        cout << endl;

        file << "Name: " << data[i].name << "\nHealth: " << data[i].health

             << "\nAttack Power: " << data[i].attack << "\nDefense Power: " << data[i].defense << "\n\n";

    }

    cout << "Data was saved in the file successfully." << endl;

    file.close(); // closing the file

    ifstream input("data.txt");

    if (!input) {

        cout << "Error: Unable to open file for reading." << endl;

        return 1;

    }

    string line;

    int a, i = 0, j;

    while (i < num) {

        j = 0;

        while (input >> a) {

            if (j == 4) continue; // skipping invalid index

            if (j == 0) data[i].name = a;

            else if (j == 1) data[i].health = a;

            else if (j == 2) data[i].attack = a;

            else if (j == 3) data[i].defense = a;

            j++;

        }

        i++;

    }

    input.close();

    ofstream write("data.txt", ios::app);

    cout << "\n\nBattle Begins: " << data[0].name << " vs " << data[1].name << endl;

    write << "\n\nBattle Begins: " << data[0].name << " vs " << data[1].name << endl;

    int s;

    int health1, health2, attack1, attack2, defense1, defense2;

    string name1, name2;

    name1 = data[0].name;

    name2 = data[1].name;

    health1 = data[0].health;

    health2 = data[1].health;

    attack1 = data[0].attack;

    attack2 = data[1].attack;

    defense1 = data[0].defense;

    defense2 = data[1].defense;

    int damage1, damage2;

    s = 0;

    while (health1 > 0 && health2 > 0) {

        if (s % 2 == 0) {

            damage2 = attack1 - defense2;

            health2 = max(0, health2 - damage2);

            cout << data[0].name << " attacks " << data[1].name << " and deals " << damage2 << " damage. "

                 << data[1].name << "'s health: " << health2 << endl;

            write << data[0].name << " attacks " << data[1].name << " and deals " << damage2 << " damage. "

                  << data[1].name << "'s health: " << health2 << endl;

        } else {

            damage1 = attack2 - defense1;

            health1 = max(0, health1 - damage1);

            cout << data[1].name << " attacks " << data[0].name << " and deals " << damage1 << " damage. "

                 << data[0].name << "'s health: " << health1 << endl;

            write << data[1].name << " attacks " << data[0].name << " and deals " << damage1 << " damage. "

                  << data[0].name << "'s health: " << health1 << endl;

        }

        s++;

    }

    if (health1 == 0) {

        cout << data[1].name << " wins!\n";

        write << data[1].name << " wins!\n";

    } else if (health2 == 0) {

        cout << data[0].name << " wins!\n";

        write << data[0].name << " wins!\n";

    }

    cout << "\nData was saved in the file successfully.\n";

    write << "\nData was saved in the file successfully.\n";

    write.close();

    delete[] data;

    return 0;

}

**Problem 101:**

**A sports club is maintaining performance statistics for two players. Each player’s details include their name, sport, matches played, and total score. The club wants a system to calculate the average score for each player and display players with an average score greater than 50. Write a C++ program that defines a struct named Player with the following attributes:**

**• name, sport, matchesPlayed, totalScore, and averageScore.**

**Take input for two players, calculate their average scores, and display their details. Display players with an average score above 50.**

**Code:**

#include<iostream>

#include<string>

using namespace std;

struct player {

string name;

string sport;

int matchesplayed;

int totalscore;

double averagescore;

};

int main()

{

player p1, p2;

cout << "Enter details for Player 1:"<<endl;

cout << "NAME: ";

getline(cin, p1.name);

cout << "Sport: ";

cin >> p1.sport;

cout << "Matches Played: ";

cin >> p1.matchesplayed;

cout << "Total Score: ";

cin >> p1.totalscore;

cin.ignore();

p1.averagescore = p1.totalscore / (double)p1.matchesplayed;

cout << endl << endl;

cout << "Enter details for Player 2:" << endl;

cout << "NAME: ";

getline(cin, p2.name);

cout << "Sport: ";

cin >> p2.sport;

cout << "Matches Played: ";

cin >> p2.matchesplayed;

cout << "Total Score: ";

cin >> p2.totalscore;

cin.ignore();

p2.averagescore = p2.totalscore / (double) p2.matchesplayed;

cout << endl << endl;

cout << "Player Details:" << endl<<endl;

cout << "Player 1 - Name: " << p1.name << ", Sport: " << p1.sport << ",

Matches Played: " << p1.matchesplayed

<< ", Average Score: " << p1.averagescore<<endl<<endl;

cout << "Player 2 - Name: " << p2.name << ", Sport: " << p2.sport << ",

Matches Played: " << p2.matchesplayed

<< ", Average Score: " << p2.averagescore << endl << endl;

cout << "Players with an average score above 50:"<<endl;

if (p1.averagescore > 50)

cout << p1.name << " with an Average Score of " <<

p1.averagescore<<endl;

if (p2.averagescore > 50)

cout << p2.name << " with an Average Score of " << p2.averagescore <<

endl;

return 0;

}

**Problem 102:**

Write a **C++ program** that:

1. Takes **two positive integers** as input, representing the **start** and **end** of a range.
2. Ensures that the start is **less than or equal** to the end and both numbers are **non-negative**.
3. Finds and displays all **composite numbers** in the given range.
   * A **composite number** is a number that has **more than two** factors (excluding 0 and 1).
4. Calculates and prints the **sum of all composite numbers** in the range.

**Code:**

#include<iostream>

using namespace std;

int main()

{

int start, end,c,sum=0;

cout << "Enter the start of the range: ";

cin >> start;

cout << "Enter the end of the range: ";

cin >> end;

if ((start > end)||(start<0||end<0))

{

cout << "Invalid range. Please ensure the start is less than or equal

to the end, and both are positive integers.";

return 0;

}

cout << "Composite numbers between " << start << " and " << end << " are : ";

for (int i = start; i <= end; i++)

{

c = 0;

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

{

if (i % j == 0)

{

c++;

}

}

if (c != 2||i==2)

{

if (i != 0 && i != 1)

{

cout << i << " ";

sum = sum + i;

}

}

}

cout << endl;

cout << "Sum of composite numbers: " << sum;

}

**Project**

**Electricity bill and management system**

#include<iostream>

#include<string>

#include<fstream>

#include<iomanip>

using namespace std;

int ptv = 35;

double fpa = 8.00;

double sales\_tax = 5.00;

int revenue = 0;

int UNITS = 0;

float unit\_prices[3][4] = {

    {12, 18, 25, 30},       // Residential

    {15, 25, 40, 50},       // Commerecial

    {20, 30, 45, 60}        // Industry

};

struct info

{                                                            //struct

    string category;

    string name;

    string cnic;

    string address;

    int billid = 0;

    int previous\_units = 0;

    int current\_units = 0;

    int units\_consumed = 0;

    double total\_bill = 0.0;

    string status = "unpaid";

};

//prototypes of update setting option

void current\_taxes();

void update\_taxes();

void change\_unit\_price();

void current\_unit\_price();

void updatesetting();

//protoypes of search bill option

int searchbill(info& p1);

int stringToInt(string);

int taxes(double total\_bill)             //bill calculating on taxes

{

    double fpa\_, sales\_tax\_;

    double total\_taxes;

    cout << "PTV Fee: " << ptv << " /-Rs" << endl;

    fpa\_ = total\_bill \* (fpa / 100);

    cout << "Fuel Price Adjustment (" << fpa << "%):" << fpa\_ << " Rs" << endl;

    sales\_tax\_ = total\_bill \* (sales\_tax / 100);

    cout << "Sales Tax (" << sales\_tax << "%):" << sales\_tax\_ << " Rs" << endl;

    total\_taxes = ptv + fpa\_ + sales\_tax\_;

    cout << "Total amount of taxes: " << total\_taxes << endl;

    total\_bill = total\_bill + total\_taxes;

    return total\_bill;

}

int calculatebill(int units, string category) // calculate bill on conditions like units

{                                             //consumed, unit price and category

    int u\_price;

    double total\_bill;

    if (category == "Residential")

    {

        if (units >= 0 && units <= 200)

        {

            u\_price = unit\_prices[0][0];

        }

        else if (units > 200 && units <= 500)

        {                                               //Residential

            u\_price = unit\_prices[0][1];

        }

        else if (units > 500 && units <= 1000)

        {

            u\_price = unit\_prices[0][2];

        }

        else

        {

            u\_price = unit\_prices[0][3];

        }

    }

    else if (category == "Commercial")

    {

        if (units >= 0 && units <= 300)

        {

            u\_price = unit\_prices[1][0];

        }

        else if (units > 300 && units <= 700)

        {

            u\_price = unit\_prices[1][1];              //Commercial

        }

        else if (units > 700 && units <= 1200)

        {

            u\_price = unit\_prices[1][2];

        }

        else

        {

            u\_price = unit\_prices[1][3];

        }

    }

    else

    {

        if (units >= 0 && units <= 500)

        {

            u\_price = unit\_prices[2][0];

        }

        else if (units > 500 && units <= 1000)

        {                                                //Industry

            u\_price = unit\_prices[2][1];

        }

        else if (units > 1000 && units <= 2000)

        {

            u\_price = unit\_prices[2][2];

        }

        else

        {

            u\_price = unit\_prices[2][3];

        }

    }

    total\_bill = u\_price \* units;

    return total\_bill;

}

int unitsconsumed(info& p1)                        // units consumed

{

    ofstream file("billsdata.txt", ios::app);

    if (!file)

    {

        cout << "Error: Opening the file on third time.";

        return 1;

    }

    int cur, prev, units;

    do {

        cout << "Previous Reading: ";

        cin >> prev;

    } while (prev < 0);

    do {

        cout << "Current Reading: ";

        cin >> cur;

    } while (cur < prev);

    p1.previous\_units = prev;

    file << setw(10) << p1.previous\_units;

    p1.current\_units = cur;

    file << setw(15) << p1.current\_units;

    units = cur - prev;            //globally store the total units provided by company

    UNITS += units;

    p1.units\_consumed = units;

    file << setw(15) << p1.units\_consumed;

    file.close();

    return units;

}

info customerdetails(info& p1)                      //customer details

{

    cin.ignore();

    ofstream file("billsdata.txt", ios::app);

    if (!file)

    {

        cout << "Error: Opening the file on second time.";

        exit(1);

    }

    p1.billid = rand();

    file << left << setw(10) << p1.billid;

    cout << "Bill ID: " << p1.billid << endl;

    file << setw(15) << p1.status;

    cout << "Enter the Name: ";

    getline(cin, p1.name);

    file << setw(15) << p1.name;

    cout << "Enter the Address: ";

    getline(cin, p1.address);

    file << setw(40) << p1.address;

    cout << "Enter CNIC #: ";

    cin >> p1.cnic;

    file << setw(12) << p1.cnic;

    cin.ignore();

    file.close();

    return p1;

}

void generatebill(info& p1)         //  bill generating

{

    int units;

    double total\_bill;

    customerdetails(p1);

    cout << endl;

    units = unitsconsumed(p1);

    cout << endl;

    cout << "No. of Units consumed: " << units;

    total\_bill = calculatebill(units, p1.category);

    cout << endl;

    cout << "------------Bill Without Taxes----------------" << endl;

    cout << "Total Bill without Taxes: " << total\_bill << " Rs" << endl;

    cout << "------------Bill After Taxes------------------" << endl;

    total\_bill = taxes(total\_bill);

    ofstream file("billsdata.txt", ios::app);

    if (!file)

    {

        cout << "Error: Opening the file on fourth time.";

        return;

    }

    p1.total\_bill = total\_bill;

    file << setw(22) << p1.total\_bill;

    file << setw(20) << p1.category << "\n";

    file.close();

    cout << "Total bill after taxes: " << total\_bill << " Rs";

}

int main()

{

    ofstream file("billsdata.txt");

    if (!file)

    {

        cout << "Error: Opening the file on first time.";

        return 1;

    }

    file << left << setw(10) << "Bill-ID" << setw(15) << "Status" << setw(15) << "Name" << setw(40)

        << "Address" << setw(17) << "CNIC#" << setw(16) << "Prev\_U" << setw(15) << "Current\_U"

        << setw(15) << "Consumed\_U" << setw(20) << "Total Bill" << "Category\n";

    file.close();

    info p1;

    srand(time(0));

    cout << "" << endl;       //formating

    cout << "                ==============================================================" << endl << endl;

    cout << "                |"; cout << "\033[1;35m"; cout << "           Najmul Arifeen | Roll No : FA24 - BSE - 102"; cout << "\033[0m"; cout << "        | " << endl << endl;

    cout << "                |"; cout << "\033[1;35m"; cout << "     Muhammad Hasnat Imtiaz | Roll No : FA24 - BSE - 081"; cout << "\033[0m"; cout << "      | " << endl << endl;

    cout << "                 ===============================================================            " << endl << endl << endl;

    cout << "              ==============================================================================" << endl << endl;

    cout << "              \*"; cout << "\033[1;34m"; cout << "                            ELECTRICITY BILL MANAGMENT SYSTEM \* " << endl << endl;

    cout << "\033[0m"; // Reset color

    cout << "              ==============================================================================" << endl;;

    cout << "" << endl << endl;

    string option[] = {

    "1. Calculate Electricity Bill\n",

    "2. Adjust Setting\n",

    "3. Search Bill\n",

    "4. Total Revenue and Consumed Units\n",

    "5. EXIT\n"

    };

    string category;

    int choice, num;

    cout << "==========================" << endl;

    cout << " Please Select an option " << endl;

    cout << "==========================" << endl << endl;

    while (true)

    {

        cout << "\033[1;31m";//red color

        for (int i = 0; i < 5; i++) {

            cout << option[i];

        }

        cout << "\033[0m";//reset color

        cout << endl;

        cout << "Enter your choice: ";

        cin >> choice;

        switch (choice) {

        case 1:

            for (int i = 1; i <= 3; i++)

            {

                if (i == 1)

                {

                    cout << endl;

                    cout << "Category ----->>>  Residential" << endl;

                    p1.category = "Residential";

                    cout << "Enter the number of residencial meter: ";

                    cin >> num;

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

                    {

                        generatebill(p1);

                        cout << endl << endl;

                    }

                    cout << endl;

                }

                else if (i == 2)

                {

                    cout << endl;

                    cout << "Category ----->>>  Commercial" << endl;

                    p1.category = "Commercial";

                    cout << "Enter the number of commercial meter: ";

                    cin >> num;

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

                    {

                        generatebill(p1);

                        cout << endl << endl;

                    }

                    cout << endl;

                }

                else if (i == 3)

                {

                    cout << endl;

                    cout << "Category ----->>>  Industry" << endl;

                    p1.category = "Industry";

                    cout << "Enter the number of industrial meter: ";

                    cin >> num;

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

                    {

                        generatebill(p1);

                        cout << endl << endl;

                    }

                    cout << endl;

                }

                cout << endl;

            }

            break;

        case 2:

        {

            updatesetting();

            break;

        }

        case 3:

        {

            char ch;

            do {

                searchbill(p1);

                cout << "Do you want to continue searching (Y/N)";

                cin >> ch;

                if (ch == 'y' || ch == 'Y')

                {

                    continue;

                }

                else

                {

                    cout << "Exiting from searchbill option.\n\n";

                    break;

                }

            } while (true);

            break;

        }

        case 4:

            cout << "Total revenue =======>>>>   " << revenue << "/-Rs" << endl;

            cout << "Total units consumed ======>>>>>>   " << UNITS << endl;

            break;

        case 5:

            return 0;

            break;

        default:

            cout << "Wrong Option!\n\n";

            break;

        }

        cout << endl;

    }

    return 0;

}

void current\_taxes()              //current taxes

{

    cout << "PTV Fee (in rupees): " << ptv << "Rs" << endl;

    cout << "Fuel Price Adjustment: " << fpa << "%" << endl;

    cout << "Sales Tax: " << sales\_tax << "%" << endl << endl;

}

void update\_taxes()              //update taxes

{

    cout << "Enter the PTV fee (in rupees): ";

    cin >> ptv;

    cout << "Enter the Fuel Price Adjustment (in percentage): ";

    cin >> fpa;

    cout << "Enter the Sales Tax (in percentage): ";

    cin >> sales\_tax;

    cout << endl;

}

void change\_unit\_price()                      //changing the unit price

{

    cout << "------Residential------" << endl << endl;

    cout << " 0-200: ";

    cin >> unit\_prices[0][0];

    cout << " 201-500: ";

    cin >> unit\_prices[0][1];

    cout << " 501-1000: ";

    cin >> unit\_prices[0][2];

    cout << " Greater than 1000: ";

    cin >> unit\_prices[0][3];

    cout << endl;

    cout << "------Commercial------" << endl << endl;

    cout << " 0-300: ";

    cin >> unit\_prices[1][0];

    cout << " 301-700: ";

    cin >> unit\_prices[1][1];

    cout << " 701-1200: ";

    cin >> unit\_prices[1][2];

    cout << " Greater than 1200: ";

    cin >> unit\_prices[1][3];

    cout << endl;

    cout << "------Industry------" << endl << endl;

    cout << " 0-500: ";

    cin >> unit\_prices[2][0];

    cout << " 501-1000: ";

    cin >> unit\_prices[2][1];

    cout << " 1001-2000: ";

    cin >> unit\_prices[2][2];

    cout << " Greater than 2000: ";

    cin >> unit\_prices[2][3];

}

void current\_unit\_price()     //current unit prices according to category

{

    cout << "------Residential------" << endl << endl;

    cout << " 0-200: " << unit\_prices[0][0] << "/Rs" << endl;

    cout << " 201-500: " << unit\_prices[0][1] << "/Rs" << endl;

    cout << " 501-1000: " << unit\_prices[0][2] << "/Rs" << endl;

    cout << " Greater than 1000: " << unit\_prices[0][3] << "/Rs" << endl;

    cout << endl;

    cout << "------Commercial------" << endl << endl;

    cout << " 0-300: " << unit\_prices[1][0] << "/Rs" << endl;

    cout << " 301-700: " << unit\_prices[1][1] << "/Rs" << endl;

    cout << " 701-1200: " << unit\_prices[1][2] << "/Rs" << endl;

    cout << " Greater than 1200: " << unit\_prices[1][3] << "/Rs" << endl;

    cout << endl;

    cout << "------Industry------" << endl << endl;

    cout << " 0-500: " << unit\_prices[2][0] << "/Rs" << endl;

    cout << " 501-1000: " << unit\_prices[2][1] << "/Rs" << endl;

    cout << " 1001-2000: " << unit\_prices[2][2] << "/Rs" << endl;

    cout << " Greater than 2000: " << unit\_prices[2][3] << "/Rs" << endl << endl;

}

void updatesetting()        //updating the settings

{

    char ch;

    cout << "================Current Setting================" << endl << endl;

    cout << "----------------Current Taxes-------------------" << endl << endl;

    current\_taxes();

    cout << endl << endl;

    cout << "----------------Current Units Rate-------------------" << endl << endl;

    current\_unit\_price();

    do {

        cout << "Do you want to change the setting of taxes: (Y/N): ";

        cin >> ch;

        cout << endl;

        if (ch == 'Y' || ch == 'y')

        {

            update\_taxes();

        }

    } while (ch != 'y' && ch != 'Y' && ch != 'n' && ch != 'N');

    cout << endl;

    do {

        cout << "Do you want to change the setting of units price: (Y/N): ";

        cin >> ch;

        cout << endl;

        if (ch == 'Y' || ch == 'y')

        {

            change\_unit\_price();

        }

    } while (ch != 'y' && ch != 'Y' && ch != 'n' && ch != 'N');

    cout << "-------------------------------------" << endl;

}

int searchbill(info& p1)                // searching of bill

{

    string line, target;

    ifstream file("billsdata.txt");

    ofstream tempFile("temp.txt", ios::out);

    cout << "Enter the bill ID to search the bill: ";

    cin >> target;

    if (!file) {

        cout << "Error: Unable to open the file in searchbill function!" << endl;

        return 1;

    }

    if (!tempFile) {

        cout << "Error: Unable to create temporary file!" << endl;

        return 1;

    }

    bool found = false;

    while (getline(file, line))

    {

        string billid = "", status = "", address = "", name = "";

        string cnic = "", prev\_u = "", curr\_u = "", consumed\_u = "", category = "";

        string total\_bill = "";

        int i = 0;

        // Extract the bill-id

        while (line[i] != ' ') {

            billid += line[i];

            i++;

        }

        // Skip spaces after the bill-id

        while (line[i] == ' ') {

            i++;

        }

        // Extract the status

        while (line[i] != ' ') {

            status += line[i];

            i++;

        }

        // Skip spaces after the status

        while (line[i] == ' ') {

            i++;

        }

        // Extract the name

        while (line[i] != ' ') {

            name += line[i];

            i++;

        }

        // Skip spaces after the name

        while (line[i] == ' ') {

            i++;

        }

        // Extract the address

        while (line[i] != ' ') {

            address += line[i];

            i++;

        }

        // Skip spaces after the address

        while (line[i] == ' ') {

            i++;

        }

        // Extract the total-bill

        while (line[i] != ' ') {

            cnic += line[i];

            i++;

        }

        // Skip spaces after total-bill

        while (line[i] == ' ') {

            i++;

        }

        // Extract the cnic

        while (line[i] != ' ') {

            prev\_u += line[i];

            i++;

        }

        // Skip spaces after cnic

        while (line[i] == ' ') {

            i++;

        }

        // Extract the previous units

        while (line[i] != ' ') {

            curr\_u += line[i];

            i++;

        }

        // Skip spaces after previous units

        while (line[i] == ' ') {

            i++;

        }

        // Extract the current units

        while (line[i] != ' ') {

            consumed\_u += line[i];

            i++;

        }

        // Skip spaces after current units

        while (line[i] == ' ') {

            i++;

        }

        // Extract the total bill

        while (line[i] != ' ') {

            total\_bill += line[i];

            i++;

        }

        // Skip spaces after consumed units

        while (line[i] == ' ') {

            i++;

        }

        // Extract the category

        while (line[i] != '\0') {

            category += line[i];

            i++;

        }

        // Update status if the bill ID matches

        if (billid == target) {

            found = true;

            char ch;

            cout << "\nBill found!\n";

            cout << "\n\n-----------------Bill Details-----------------" << endl;

            cout << "Bill ID: " << billid << endl;

            cout << "Status: " << status << endl;

            cout << "Name: " << name << endl;

            cout << "Address: " << address << endl;

            cout << "CNIC: " << cnic << endl;

            cout << "Previous Units: " << prev\_u << endl;

            cout << "Current Units: " << curr\_u << endl;

            cout << "Consumed Units: " << consumed\_u << endl;

            cout << "Total Bill: " << total\_bill << endl;

            cout << "Category: " << category << endl;

            if (status == "unpaid") {

                do {

                    cout << "Do you want to pay the bill? (Y/N): ";

                    cin >> ch;

                    if (ch == 'Y' || ch == 'y') {

                        status = "paid";

                        cout << "\nStatus updated to 'paid' successfully!\n" << endl;

                        revenue = revenue + stringToInt(total\_bill);

                    }

                    else if (ch == 'N' || ch == 'n')

                    {

                        cout << "\nNo changes made to the bill status.\n" << endl;

                    }

                } while (ch != 'y' && ch != 'Y' && ch != 'n' && ch != 'N');

            }

            // Write the updated line to the temp file

            tempFile << left << setw(10) << billid << setw(15) << status << setw(15) << name << setw(40) << address

                << setw(12) << cnic << setw(10) << prev\_u << setw(15) << curr\_u << setw(15)

                << consumed\_u << setw(22) << total\_bill << setw(20) << category << "\n";

        }

        else

        {

            tempFile << line << endl;

        }

    }

    if (!found)

    {

        cout << "Bill ID not found!\n" << endl;

    }

    file.close();

    tempFile.close();                                                           // Close the files

    // Replace the original file with the temporary file

    if (remove("billsdata.txt") != 0) {

        cout << "Error: Unable to delete the original file!" << endl;

        return 1;

    }

    if (rename("temp.txt", "billsdata.txt") != 0) {

        cout << "Error: Unable to rename the temporary file!" << endl;

        return 1;

    }

    return 0;

}

int stringToInt(string str)

{

    int result = 0;

    for (char ch : str)

    {

        result = result \* 10 + (ch - '0'); // Convert character to digit and accumulate

    }

    return result;

}