**WEEK 6:**

**CODE**

#include <iostream>

#include <iomanip>

using namespace std;

// Constants for weight limits

const double MIN\_WEIGHT\_CEMENT = 24.9;

const double MAX\_WEIGHT\_CEMENT = 25.1;

const double MIN\_WEIGHT\_SAND\_GRAVEL = 49.9;

const double MAX\_WEIGHT\_SAND\_GRAVEL = 50.1;

// Prices for sacks

const double PRICE\_CEMENT = 3.0;

const double PRICE\_GRAVEL = 2.0;

const double PRICE\_SAND = 2.0;

const double PRICE\_SPECIAL\_PACK = 10.0;

// Function to check a single sack

bool checkSack(char content, double weight) {

if ((content == 'C' && weight >= MIN\_WEIGHT\_CEMENT && weight <= MAX\_WEIGHT\_CEMENT) ||

((content == 'G' || content == 'S') && weight >= MIN\_WEIGHT\_SAND\_GRAVEL && weight <= MAX\_WEIGHT\_SAND\_GRAVEL)) {

return true;

} else {

cout << "Rejected sack - ";

if (content != 'C' && content != 'G' && content != 'S') {

cout << "Invalid content; ";

}

if ((content == 'C' && (weight < MIN\_WEIGHT\_CEMENT || weight > MAX\_WEIGHT\_CEMENT)) ||

((content == 'G' || content == 'S') && (weight < MIN\_WEIGHT\_SAND\_GRAVEL || weight > MAX\_WEIGHT\_SAND\_GRAVEL))) {

cout << "Invalid weight; ";

}

cout << endl;

return false;

}

}

// Function to check a customer's order

void checkOrder(int &sacksCement, int &sacksGravel, int &sacksSand, double &totalPrice, int &rejectedSacks) {

cout << "Enter the number of sacks of cement, gravel, and sand for the order:" << endl;

cin >> sacksCement >> sacksGravel >> sacksSand;

rejectedSacks = 0;

// Check and accumulate weights and prices

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

char content;

double weight;

cout << "Enter content and weight for cement sack " << i + 1 << ": ";

cin >> content >> weight;

if (!checkSack(content, weight)) {

rejectedSacks++;

} else {

totalPrice += PRICE\_CEMENT;

}

}

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

char content;

double weight;

cout << "Enter content and weight for gravel sack " << i + 1 << ": ";

cin >> content >> weight;

if (!checkSack(content, weight)) {

rejectedSacks++;

} else {

totalPrice += PRICE\_GRAVEL;

}

}

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

char content;

double weight;

cout << "Enter content and weight for sand sack " << i + 1 << ": ";

cin >> content >> weight;

if (!checkSack(content, weight)) {

rejectedSacks++;

} else {

totalPrice += PRICE\_SAND;

}

}

}

// Function to calculate the price for a customer's order

void calculatePrice(int sacksCement, int sacksGravel, int sacksSand, double totalPrice, int rejectedSacks) {

int specialPacks = min(min(sacksCement, sacksGravel / 2), sacksSand / 2);

totalPrice -= specialPacks \* PRICE\_SPECIAL\_PACK;

cout << fixed << setprecision(2);

cout << "Regular price: $" << totalPrice << endl;

if (specialPacks > 0) {

cout << "Discount price (after applying " << specialPacks << " special pack(s)): $" << totalPrice << endl;

cout << "Amount saved: $" << specialPacks \* PRICE\_SPECIAL\_PACK << endl;

}

}

int main() {

int sacksCement, sacksGravel, sacksSand, rejectedSacks = 0;

double totalPrice = 0.0;

// Check a single sack

char content;

double weight;

cout << "Enter content and weight for a single sack:" << endl;

cin >> content >> weight;

if (checkSack(content, weight)) {

cout << "Accepted sack - Content: " << content << ", Weight: " << weight << " kg" << endl;

}

// Check a customer's order for delivery

checkOrder(sacksCement, sacksGravel, sacksSand, totalPrice, rejectedSacks);

cout << "Total weight of the order: " << fixed << setprecision(2) << totalPrice << " kg" << endl;

cout << "Number of rejected sacks: " << rejectedSacks << endl;

// Calculate the price for a customer's order

calculatePrice(sacksCement, sacksGravel, sacksSand, totalPrice, rejectedSacks);

return 0;

}

**OUTPUT:**

