Computer Science and Creative Technologies

**Task**

**Task 1** : Programming Exercises:[Data types and Conditional Statements]

1. Write a program that takes a temperature value from the user. It should then allow the user to choose between Celsius (C) and Fahrenheit (F) for conversion. After the user selection, it should then convert the entered temperature to the chosen scale and display the result.

Use appropriate data types for temperature and handle error like non-numeric input.

Use the following formula for conversion:

F = (C x 9/5) + 32

C = (F - 32) x 5/9 [10 marks]

**Answer:**

#include <iostream>

using namespace std;

class TemperatureConverter {

private:

int choice;

double temp;

double op;

public:

void showMenu() {

cout << "Press 1 for Celsius to Fahrenheit!" << endl;

cout << "Press 2 for Fahrenheit to Celsius!" << endl;

getChoice();

}

void getChoice() {

cout << "Enter your choice: ";

cin >> choice;

getTemperature();

}

void getTemperature() {

if (choice == 1) {

getInput();

convertToFahrenheit();

} else if (choice == 2) {

getInput();

convertToCelsius();

} else {

cout << "Invalid response!" << endl;

}

}

void getInput() {

cout << "Enter temperature: ";

cin >> temp;

}

void convertToCelsius() {

op = (temp - 32) \* 5 / 9;

displayResult();

}

void convertToFahrenheit() {

op = (temp \* 9.0 / 5.0) + 32;

displayResult();

}

void displayResult() {

cout << "The converted temperature is: " << op << endl;

}

};

int main() {

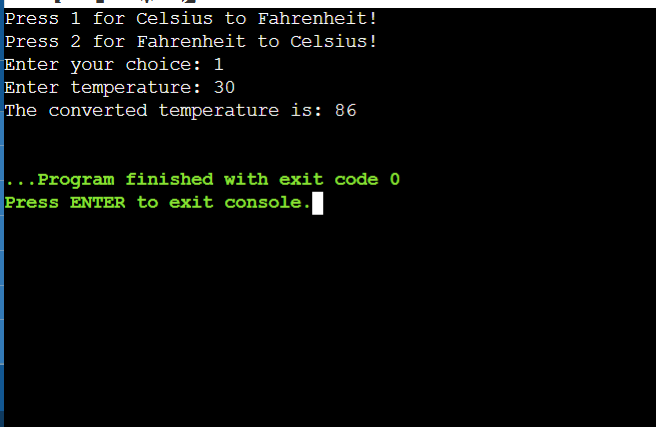
TemperatureConverter converter;

converter.showMenu();

return 0;

}

Output:



1. Write a C++ program to implement a number guessing game with different difficulty levels.

Easy difficulty ranges from 1-8, medium from 1-30, hard from 1-50.Then,generate a random number to check if the guess is correct based on the user's selection. [10 marks]

Answer:

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

class NumberGuessingGame {

private:

int maxNo;

int randomNo;

int attempts;

public:

NumberGuessingGame(int max) : maxNo(max), attempts(0) {

srand(time(0));

randomNo = rand() % maxNo + 1;

}

void play() {

int guess;

cout << "Guess the number between 1 and " << maxNo << ": ";

do {

cin >> guess;

attempts++;

if (guess < randomNo)

cout << "Too low! Try again: ";

else if (guess > randomNo)

cout << "Too high! Try again: ";

} while (guess != randomNo);

cout << "Congratulations! You guessed the number " << randomNo

<< " correctly in " << attempts << " attempts.\n";

}

};

int main() {

int choice;

cout << "Select difficulty level:\n";

cout << "1. Easy (1-8)\n";

cout << "2. Medium (1-30)\n";

cout << "3. Hard (1-50)\n";

cout << "Enter your choice: ";

cin >> choice;

int maxNo;

switch (choice) {

case 1:

maxNo = 8;

break;

case 2:

maxNo = 30;

break;

case 3:

maxNo = 50;

break;

default:

cout << "Invalid choice. Exiting program.\n";

return 1;

}

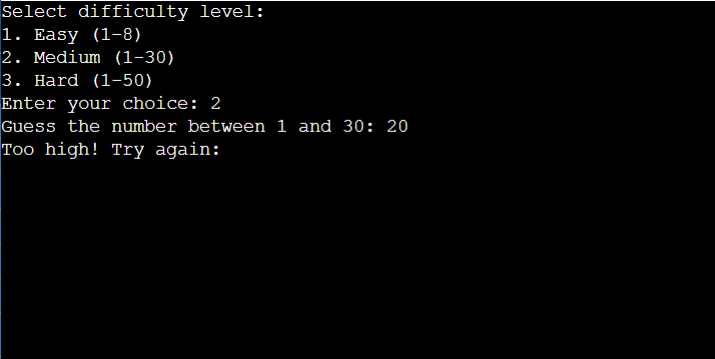
NumberGuessingGame game(maxNo);

game.play();

return 0;

}

Output:



1. Write a program that reads an array of integer numbers from the user and finds the minimum and maximum values in the entered array.

[10 marks]

Answer:

#include <iostream>

using namespace std;

class Array {

public:

Array(int size) : size(size), arr(new int[size]) {}

~Array() {

delete[] arr;

}

void readArray() {

cout << "Enter " << size << " integer numbers: ";

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

cin >> arr[i];

}

}

void findMinMax() {

int min = arr[0];

int max = arr[0];

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

if (arr[i] < min) {

min = arr[i];

}

if (arr[i] > max) {

max = arr[i];

}

}

cout << "Minimum value: " << min << endl;

cout << "Maximum value: " << max << endl;

}

private:

int size;

int\* arr;

};

int main()

{

int size;

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

cin >> size;

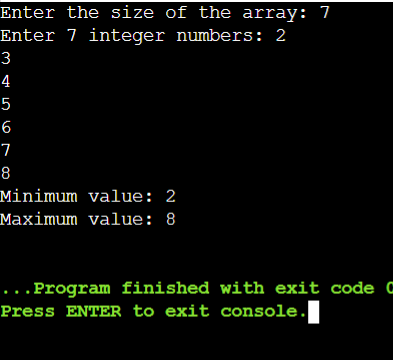
Array a1(size);

a1.readArray();

a1.findMinMax();

    return 0;

}

Output

1. Write a program that reads a number from the user and based on the user input, it says what day of the week it is, Sundays being 1 and Saturdays being 7. You system should give appropriate response for invalid input entries.

Answer:

#include <iostream>

using namespace std;

class DayOfWeek {

private:

int dayNumber;

string weekDays[7] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};

public:

void assignDay(int num) {

dayNumber = num;

}

void showDay() {

if (dayNumber >= 1 && dayNumber <= 7)

cout << "You selected: " << weekDays[dayNumber - 1] << endl;

else

cout << "Invalid choice! Please provide a value between 1 and 7." << endl;

}

};

int main() {

DayOfWeek myDay;

int input;

cout << "Please enter a number from 1 to 7 to know the day of the week: ";

cin >> input;

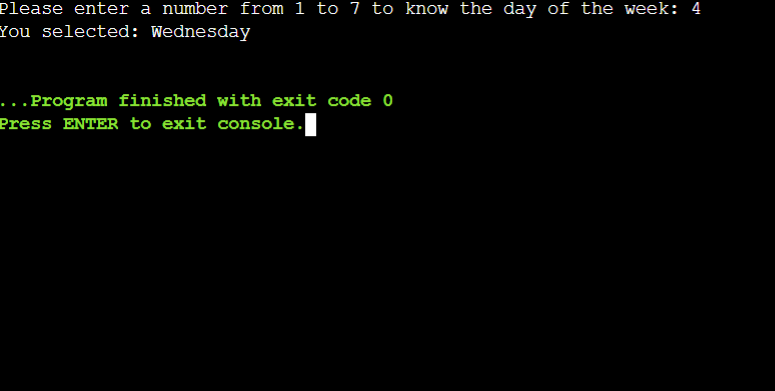
myDay.assignDay(input);

myDay.showDay();

return 0;

}

Output:



1. ⁠Create a program that takes a positive integer as input and determines whether it's a "bouncy number". A bouncy number is one where the digits neither consistently increase nor consistently decrease when read from left to right. For example:

•⁠ ⁠123 is NOT bouncy (digits consistently increase)

•⁠ ⁠321 is NOT bouncy (digits consistently decrease)

•⁠ ⁠120 is bouncy (neither consistently increasing nor decreasing)

Answer:

**#include <iostream>**

**using namespace std;**

**bool isBouncy(int number) {**

**bool increasing = false, decreasing = false;**

**int lastDigit = number % 10;**

**number /= 10;**

**while (number > 0) {**

**int currentDigit = number % 10;**

**if (currentDigit < lastDigit)**

**increasing = true;**

**else if (currentDigit > lastDigit)**

**decreasing = true;**

**lastDigit = currentDigit;**

**number /= 10;**

**if (increasing && decreasing)**

**return true; // Bouncy**

**}**

**return false; // Not bouncy**

**}**

**int main() {**

**int num;**

**cout << "Enter a positive integer: ";**

**cin >> num;**

**if (num < 0) {**

**cout << "Invalid input! Please enter a positive integer." << endl;**

**return 1;**

**}**

**if (num < 100) {**

**cout << "Numbers below 100 are not considered bouncy." << endl;**

**} else {**

**if (isBouncy(num))**

**cout << num << " is a bouncy number." << endl;**

**else**

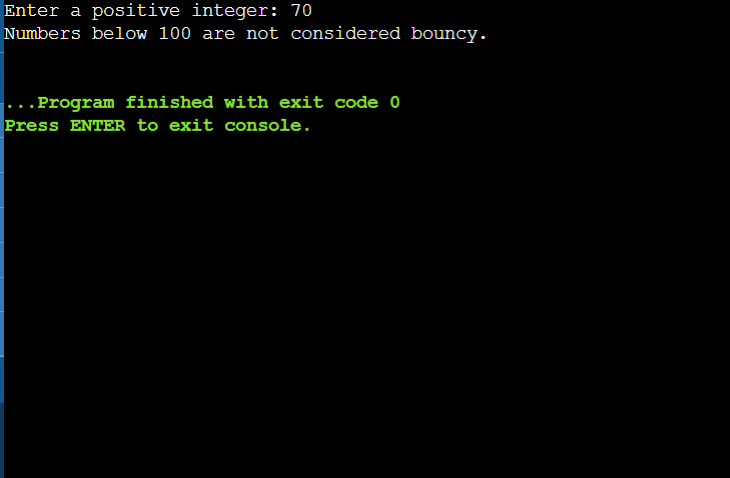
**cout << num << " is not a bouncy number." << endl;**

**}**

**return 0;**

**}**

**Output:**

****