

CS 110: Lab 04

**Repetition Statements-Loops**

**Muhammad Mujtaba**

**CMD ID: 540040**

[mmujtaba.bese25seecs@seecs.edu.pk](mailto:mmujtaba.bese25seecs@seecs.edu.pk)

**Class:** BESE 16B

**Batch:** 2k25

# Task 1 [CLO 2]: Number Guessing Game

## CODE

#include <iostream>

#include <random>

// get random value of type T (can be any type) between given values inclusive

template <typename T>

T getRandom(T minInc, T maxInc)

{

std::random\_device rd; // get a random device

std::mt19937 gen(rd()); // using a random engine

// "uniform\_int\_distribution" used to remove random bias

std::uniform\_int\_distribution<> distrib(minInc, maxInc);

// return the number

return distrib(gen);

}

int main()

{

// variables

size\_t guess\_counter = 0; // size\_t is commonly used for counters in cpp

unsigned int secret = getRandom<int>(1, 10); // not negative; <int> is not required but used to emphasize that template function is used

unsigned int guess; // not negative

// MAIN LOOP

std::cout << "I have chosen a number between 1 and 10. Try to guess it.\n\n";

do // do while is used to intialize the guess of user

{

std::cout << "Your guess: ";

std::cin >> guess;

// newline by using pressing enter

// increase stored guess counter

guess\_counter++;

// if guessed correctly then print and exit

if (secret == guess)

{

std::cout << "\n\nThat's right! You guessed it. \n";

std::cout << "It only took you " << guess\_counter << " tries.\n";

break;

}

// if not executed -> guess is wrong

std::cout << "\nThat is incorrect! Guess Again.\n";

} while (secret != guess);

// ignoring previous input

std::cin.ignore();

std::cin.get();

return 0;

}

## OUTPUT



# Task 2 [CLO 2]: Computing Harmonic Mean

## CODE

#include <iostream>

int main()

{

int input\_n = 0;

// using double for more accuracy

double result\_numerator = 0.f;

double result\_denominator = 0.f;

std::cout << "Enter the amount of inputs: ";

std::cin >> input\_n;

// newline by user pressing enter

result\_numerator = input\_n;

// take `input\_n` values from user

int counter = 1;

do

{

float x\_i;

std::cout << "Enter x" << counter << ": ";

std::cin >> x\_i;

// newline by user pressing enter

result\_denominator += 1.f / x\_i;

counter++;

} while (counter <= input\_n);

std::cout << "Harmonic mean if your input numbers is " << result\_numerator / result\_denominator << "\n";

// ignoring previous input

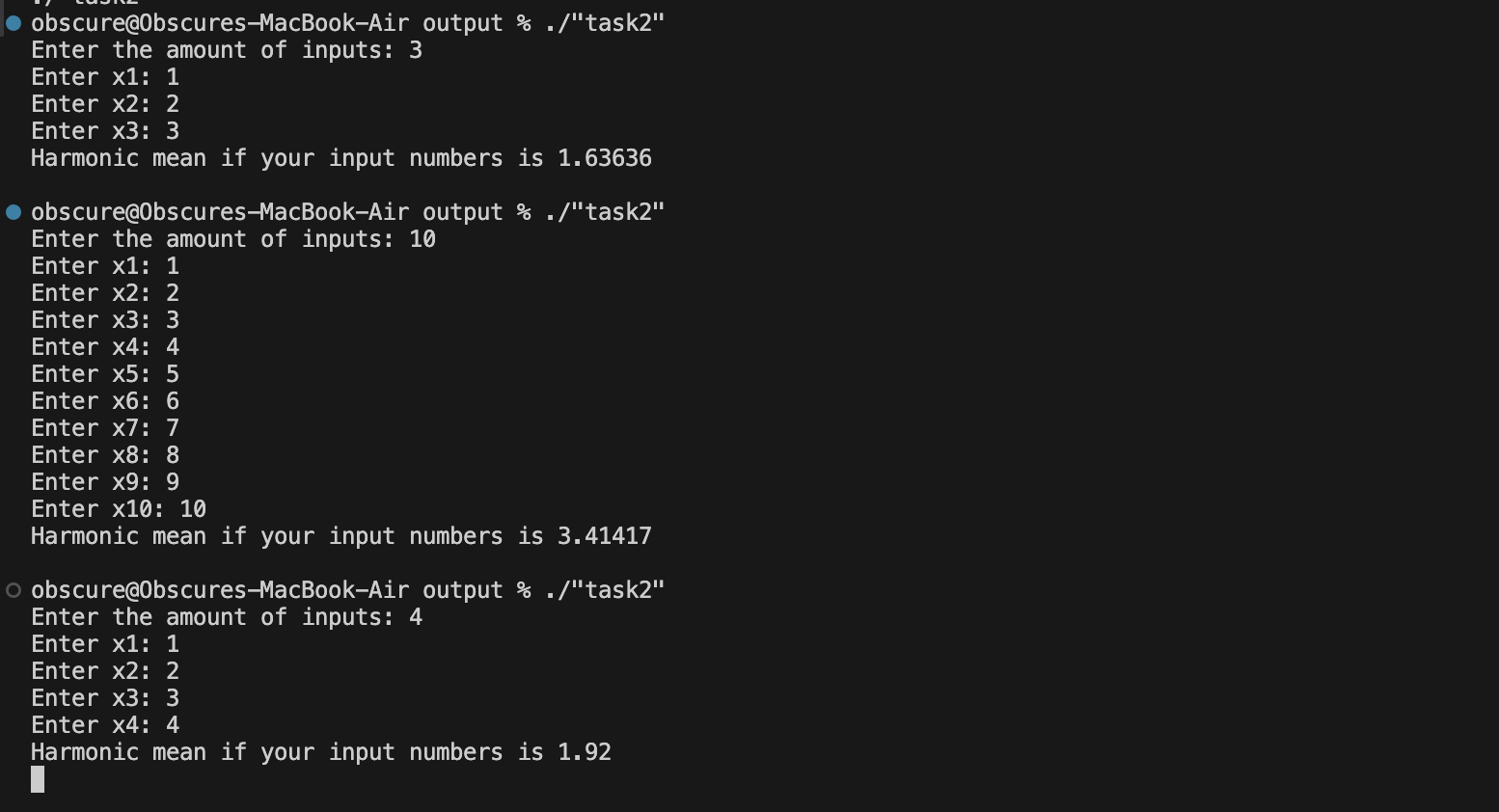
std::cin.ignore();

std::cin.get();

return 0;

}

## OUTPUT



# Task 3 [CLO 3] Factorial:

## CODE

#include <iostream>

int main()

{

int input\_n = 0;

int factorial = 1; // intializing to 1 instead of 0

std::cout << "Input a number to find factorial of: ";

std::cin >> input\_n;

// newline by user pressing enter

// if user user inputs 0 program doesn't run and returns stored value of `factorial` i.e 1

// start from 1 and multulpy all numbers till `input\_n`; store result in `factorial`

for (int i = 1; i <= input\_n; i++)

{

factorial \*= i;

}

std::cout << "Factorial of " << input\_n << " is " << factorial << "\n";

// ignoring previous input

std::cin.ignore();

std::cin.get();

return 0;

}

## OUTPUT

A screenshot of a computer program

AI-generated content may be incorrect.

# Task 4 [CLO 3] Find lowest priced item:

## CODE

#include <iostream>

#include <string>

#include <limits>

#include <iomanip>

int main()

{

int num\_items = 0;

std::string lowest\_name = "";

// float is not a simple type like int so we cannot just flip all bits to one, that is NaN

float lowest\_price = std::numeric\_limits<float>::max(); // storing max value that can be stored in float

std::cout << "Number of items to Enter: ";

std::cin >> num\_items;

std::cout << "\n";

std::cout << std::setw(32) << std::setfill('-') << "\n";

std::cout << "\n";

for (int i = 1; i <= num\_items; i++)

{

std::string name = "";

float price = 0;

std::cout << i << ". Enter the Item Name: ";

std::cin.ignore(); // ignoring previous input

std::getline(std::cin, name);

// getline reads until newline from the given stream (std::cin in this case) and stores it in the given strings

std::cout << i << ". Enter the Item Price: ";

std::cin >> price;

// newline by user pressing enter

// compare to already stored lowest value, if this is lower then update lowest

// no need to compare names

if (price <= lowest\_price)

{

lowest\_name = name;

lowest\_price = price;

}

// spacer for next item

std::cout << std::endl;

}

std::cout << "\n";

std::cout << std::setw(32) << std::setfill('-') << "\n";

std::cout << "\n";

std::cout << "Name of lowest Item: " << lowest\_name << "\n";

std::cout << "Price of lowest Item: " << lowest\_price << "\n";

// ignoring previous input

std::cin.ignore();

std::cin.get();

return 0;

}

## OUTPUT

