**Take Home Program 6 - Due on or before Sunday 11/11**

**Objective:** Objects and classes: static members

|  |
| --- |
| **Important instructions:**   * *All programs must include comments at the top of your program: your name, course name-section number (e.g. CSIT 839 -26953), program name and the program description in brief.* * *Copy and paste your program code and outputs in Part B of each program.* * *Once it is done, save and submit this word file via Canvas.* |

**1. CheckWriting.cpp**

You are given the class Numbers that can be used to translate whole dollar amounts in the range 0 through 9999 into an English description of the number. For example: the number 713 would be translated into the string seven hundred thirteen, and 8203 would be translated into the string eight thousand two hundred three. The class Numbers is as below:

class Numbers

{

public:

const static string lessThan20[]; // store number from 0 to 19

const static string tens[]; // store number as string for ten, twenty … ninety

const static string hundred;

const static string thousand;

void print (); // to print the English description

Numbers (int x) {number = x;}

private:

int number;

};

Write a program that asks the user to enter a number in the proper range and then prints out its English description. The program will terminate if a negative number was entered.

**Hints:** Use modulus (%) to find the remainder

Pseudocode of main program: {Ask user a number; Create a Numbers object; Call print function;}

**Sample Output:**

This program translates whole dollar amounts into words for the for the purpose of writing checks.

Entering a negative terminates the program.

Enter an amount for be translated into words: 8023

eight thousand twenty three

Enter another number: 5609

five thousand six hundred nine

Enter another number: 909

nine hundred nine

Enter another number: 743

seven hundred forty three

Enter another number: -9

**Copy and paste your program (source) code and the outputs after this line**

**+++++++++++++++++++++++++++++++++++++++++++++++++**

/\*

Inola Cohen

CheckWriting.cpp

Co Sci 839 - 26953

Purpose: to translate numbers into their respective words

\*/

//#include “stdafx.h”

#include <iostream>

#include <string>

using namespace std;

class Numbers

{

private:

int number;

public:

string lessThanTwenty[20] = { " ", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen" };

string tens[10] = { " ", "Ten", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninty" };

string hundred = "Hundred";

string thousand = "Thousand";

void print(); // To print the English description

Numbers(int x) {

number = x;

}

};

int main()

{

int num;

cout << "This program translates whole dollar amounts"

<< " into words for the purpose of writing checks." << endl;

cout << "Enter an amount to be translated into words: ";

cin >> num; // Get user input

Numbers number(num); // Create object

number.print(); // Call print function and display to user

cout << endl;

while (num > 0) // Continue getting input until negative number is inputted

{

cout << "Enter another number: ";

cin >> num;

Numbers number(num);

number.print();

cout << endl;

}

return 0;

}

void Numbers::print()

{

if (number > 999) // Getting thousand string

{

cout << lessThanTwenty[(number / 1000)] << " " << thousand << " ";

number %= 1000;

}

if (number > 99 && number < 1000) // Getting hundred string

{

cout << lessThanTwenty[number / 100] << " " << hundred << " ";

number %= 100;

}

if (number >= 20) // Getting tens string if it is (10, 20, 30, etc.)

{

cout << tens[(number / 10)] << " ";

number %= 10;

}

else if (number >= 10 && number < 20) // Getting strings between 10-20 (not including 10 and 20)

{

cout << lessThanTwenty[(number)] << " ";

}

if (number > 0 && number < 10) // Getting ones strings (1-9)

{

cout << lessThanTwenty[number] << " ";

}

}

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**