Introduction

Game Title: Code Breaker

Being fascinated by the ability family members or strangers to crack codes so effortlessly to the point of admiration was a motivation to pursuit building this mind number game. anybody who strives to obtain and sharpen the skill of cracking codes can play and have fun with this game. In order to train such a skill a reconstruction of the game Mastermind, made by Invicta,

So, what is Code Breaker? It is a number logic game that arrange number sequences together to generate a code only known to the opponent or a computer system. In addition to the fun part, many benefits will come from playing this game. To start, this game again confirms the rule of practice makes perfect because the more you play this game, the better logic and thinking skills get. The emotional, social, and mental reward is very important in playing this game; it aids with bettering self-esteem, mathematical abilities, especially arithmetic skills like sequences.

At the competitive gamers level, strategy becomes very important element in this game. Every number will be considered crucial and well calculated. This strategy is a useful skill in life and work place.

These aspects will certainly translate into the computer code by giving this game the versatility and the ability to expand capability and options to users. This game can be portable everywhere in everyone’s pocket. With network technologies, this game can be at every corner and every environment of our life. Digital Code Breaker brings fun and skill to everyone who has a phone or a smart device.

This game can be played everywhere, on the go, or train commute, lunch break, or at the park while your kids a playing on the swings.

Summary

Project size: 172 lines

Description lines: 32 lines

Code lines: 165

Variables around roughly 13

Library: #include <iostream, cstdlib, iomanip, ctime>; .h file was attempted to

Improve performance, but was unsuccessful. Function prototype were used instead.

Function: around 6

This program includes many concepts learned in the lecture as it contains operators and functions that were covered in the previous lessons; however, this code was built with expansion and diversity in mind. This project will include room for adding new concepts and methods like more scoring options, arrays, and may be visual effects and even sounds.

This project is difficult and challenging yet informative and rewarding; it offered a new experience to expand knowledge and test ability for attempting new challenges. This code is only a starting point for getting to discovering coding and the limitless boundaries of computer code as it brings an empowering experience.

Description:

This game is going to consist in reading headings and instructions, as well as prompts to enter digits of a certain range to match the generated code. At this stage, no restrictions in attempts will be implemented, but an attempt counter will display at the end of the game to show how many attempts it took to crack the code. This game will be scoreless also, and will be subject to project 2.

Flow Chart:

Cross reference for Project 1

|  |  |  |  |
| --- | --- | --- | --- |
| Chapter | Section | Topic | Where in code/Line # |
| Chapter 2 | 2 | The cout object |  |
|  | 3 | #include directive |  |
|  | 4 | Variables |  |
|  | 5 | Identifiers |  |
|  | 6 | Integer data type |  |
|  | 7 | The char data type |  |
|  | 8 | The C++ sting data |  |
|  | 9 | Floating point data |  |
|  | 10 | The bool data type |  |
|  | 11 | Determining size |  |
|  | 12 | Variable assignment |  |
|  | 13 | Scope |  |
|  | 14 | Arithmetic operators |  |
|  | 15 | Comments |  |
|  | 16 | Named constants |  |
|  | 17 | Programming style |  |
|  |  |  |  |
| Chapter 3 | 1 | The cin object |  |
|  | 2 | Mathematical expres |  |
|  | 3 | Mixing variables |  |
|  | 4 | Overflow/underflow |  |
|  | 5 | Type casting |  |
|  | 6 | Multiple assignment |  |
|  | 7 | Formatting output |  |
|  | 8 | Objects |  |
|  | 9 | Mathematical library |  |
|  | 10 | Debugging |  |
|  | 11 | Problem solving |  |
| Chapter 4 | 1 | Relational operators |  |
|  | 2 | If statement |  |
|  | 3 | Expanding if |  |
|  | 4 | The if/else |  |
|  | 5 | Nested if |  |
|  | 6 | If/else if statement |  |
|  | 7 | Flags |  |
|  | 8 | Logical operator |  |
|  | 9 | Checking numeric |  |
|  | 10 | Menue |  |
|  | 11 | Validating input |  |
|  | 12 | Compring char |  |
|  | 13 | conditional |  |
|  | 14f | Switch statement |  |
|  | 15 | Block variable |  |
|  |  |  |  |
| Chapter 5 | 1 | Increment/decrement |  |
|  | 2 | Introduction to loops |  |
|  | 3 | While loop |  |
|  | 4 | Counters |  |
|  | 5 | Do while loop |  |
|  | 6 | The for loop |  |
|  | 7 | Running totail |  |
|  | 8 | Sentinels |  |
|  | 9 | Deciding whick loop |  |
|  | 10 | Nested loop |  |
|  | 11 | Using files |  |
|  | 12 | Breaking |  |
|  |  |  |  |
| Chapter 6 | 1 | Modular program |  |
|  | 2 | Calling functions |  |
|  | 3 | Prototype |  |
|  | 4 | Sending data function |  |
|  | 5 | Passing data |  |
|  | 6 | Using function |  |
|  | 7 | Return statement |  |
|  | 8 | Returning value |  |
|  | 9 | Boolean value |  |
|  | 10 | Local/global variable |  |
|  | 11 | Static local |  |
|  | 12 | Default argument |  |
|  | 13 | Using reference |  |
|  | 14 | overloading function |  |
|  | 15 | Exit function |  |
|  | 16 | Stubs and drivers |  |