# C# Daily Programming Exam Tasks (Days 1–365)

## Day 1

Day 1: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 2

Day 2: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 3

Day 3: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 4

Day 4: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 5

Day 5: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 6

Day 6: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 7

Day 7: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 8

Day 8: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 9

Day 9: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 10

Day 10: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 11

Day 11: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 12

Day 12: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 13

Day 13: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 14

Day 14: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 15

Day 15: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 16

Day 16: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 17

Day 17: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 18

Day 18: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 19

Day 19: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 20

Day 20: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 21

Day 21: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 22

Day 22: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 23

Day 23: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 24

Day 24: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 25

Day 25: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 26

Day 26: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 27

Day 27: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 28

Day 28: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 29

Day 29: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 30

Day 30: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 31

Day 31: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 32

Day 32: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 33

Day 33: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 34

Day 34: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 35

Day 35: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 36

Day 36: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 37

Day 37: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 38

Day 38: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 39

Day 39: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 40

Day 40: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 41

Day 41: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 42

Day 42: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 43

Day 43: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 44

Day 44: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 45

Day 45: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 46

Day 46: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 47

Day 47: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 48

Day 48: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 49

Day 49: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 50

Day 50: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 51

Day 51: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 52

Day 52: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 53

Day 53: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 54

Day 54: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 55

Day 55: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 56

Day 56: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 57

Day 57: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 58

Day 58: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 59

Day 59: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 60

Day 60: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 61

Day 61: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 62

Day 62: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 63

Day 63: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 64

Day 64: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 65

Day 65: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 66

Day 66: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 67

Day 67: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 68

Day 68: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 69

Day 69: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 70

Day 70: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 71

Day 71: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 72

Day 72: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 73

Day 73: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 74

Day 74: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 75

Day 75: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 76

Day 76: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 77

Day 77: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 78

Day 78: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 79

Day 79: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 80

Day 80: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 81

Day 81: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 82

Day 82: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 83

Day 83: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 84

Day 84: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 85

Day 85: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 86

Day 86: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 87

Day 87: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 88

Day 88: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 89

Day 89: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 90

Day 90: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 91

Day 91: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 92

Day 92: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 93

Day 93: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 94

Day 94: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 95

Day 95: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 96

Day 96: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 97

Day 97: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 98

Day 98: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 99

Day 99: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 100

Day 100: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 101

Day 101: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 102

Day 102: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 103

Day 103: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 104

Day 104: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 105

Day 105: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 106

Day 106: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 107

Day 107: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 108

Day 108: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 109

Day 109: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 110

Day 110: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 111

Day 111: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 112

Day 112: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 113

Day 113: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 114

Day 114: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 115

Day 115: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 116

Day 116: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 117

Day 117: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 118

Day 118: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 119

Day 119: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 120

Day 120: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 121

Day 121: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 122

Day 122: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 123

Day 123: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 124

Day 124: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 125

Day 125: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 126

Day 126: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 127

Day 127: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 128

Day 128: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 129

Day 129: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 130

Day 130: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 131

Day 131: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 132

Day 132: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 133

Day 133: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 134

Day 134: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 135

Day 135: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 136

Day 136: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 137

Day 137: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 138

Day 138: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 139

Day 139: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 140

Day 140: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 141

Day 141: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 142

Day 142: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 143

Day 143: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 144

Day 144: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 145

Day 145: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 146

Day 146: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 147

Day 147: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 148

Day 148: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 149

Day 149: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 150

Day 150: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 151

Day 151: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 152

Day 152: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 153

Day 153: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 154

Day 154: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 155

Day 155: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 156

Day 156: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 157

Day 157: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 158

Day 158: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 159

Day 159: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 160

Day 160: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 161

Day 161: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 162

Day 162: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 163

Day 163: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 164

Day 164: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 165

Day 165: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 166

Day 166: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 167

Day 167: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 168

Day 168: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 169

Day 169: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 170

Day 170: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 171

Day 171: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 172

Day 172: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 173

Day 173: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 174

Day 174: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 175

Day 175: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 176

Day 176: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 177

Day 177: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 178

Day 178: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 179

Day 179: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 180

Day 180: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 181

Day 181: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 182

Day 182: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 183

Day 183: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 184

Day 184: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 185

Day 185: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 186

Day 186: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 187

Day 187: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 188

Day 188: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 189

Day 189: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 190

Day 190: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 191

Day 191: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 192

Day 192: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 193

Day 193: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 194

Day 194: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 195

Day 195: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 196

Day 196: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 197

Day 197: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 198

Day 198: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 199

Day 199: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 200

Day 200: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 201

Day 201: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 202

Day 202: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 203

Day 203: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 204

Day 204: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 205

Day 205: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 206

Day 206: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 207

Day 207: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 208

Day 208: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 209

Day 209: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 210

Day 210: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 211

Day 211: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 212

Day 212: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 213

Day 213: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 214

Day 214: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 215

Day 215: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 216

Day 216: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 217

Day 217: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 218

Day 218: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 219

Day 219: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 220

Day 220: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 221

Day 221: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 222

Day 222: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 223

Day 223: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 224

Day 224: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 225

Day 225: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 226

Day 226: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 227

Day 227: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 228

Day 228: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 229

Day 229: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 230

Day 230: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 231

Day 231: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 232

Day 232: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 233

Day 233: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 234

Day 234: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 235

Day 235: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 236

Day 236: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 237

Day 237: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 238

Day 238: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 239

Day 239: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 240

Day 240: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 241

Day 241: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 242

Day 242: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 243

Day 243: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 244

Day 244: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 245

Day 245: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 246

Day 246: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 247

Day 247: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 248

Day 248: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 249

Day 249: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 250

Day 250: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 251

Day 251: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 252

Day 252: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 253

Day 253: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 254

Day 254: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 255

Day 255: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 256

Day 256: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 257

Day 257: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 258

Day 258: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 259

Day 259: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 260

Day 260: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 261

Day 261: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 262

Day 262: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 263

Day 263: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 264

Day 264: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 265

Day 265: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 266

Day 266: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 267

Day 267: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 268

Day 268: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 269

Day 269: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 270

Day 270: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 271

Day 271: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 272

Day 272: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 273

Day 273: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 274

Day 274: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 275

Day 275: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 276

Day 276: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 277

Day 277: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 278

Day 278: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 279

Day 279: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 280

Day 280: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 281

Day 281: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 282

Day 282: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 283

Day 283: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 284

Day 284: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 285

Day 285: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 286

Day 286: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 287

Day 287: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 288

Day 288: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 289

Day 289: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 290

Day 290: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 291

Day 291: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 292

Day 292: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 293

Day 293: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 294

Day 294: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 295

Day 295: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 296

Day 296: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 297

Day 297: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 298

Day 298: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 299

Day 299: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 300

Day 300: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 301

Day 301: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 302

Day 302: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 303

Day 303: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 304

Day 304: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 305

Day 305: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 306

Day 306: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 307

Day 307: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 308

Day 308: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 309

Day 309: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 310

Day 310: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 311

Day 311: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 312

Day 312: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 313

Day 313: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 314

Day 314: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 315

Day 315: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 316

Day 316: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 317

Day 317: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 318

Day 318: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 319

Day 319: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 320

Day 320: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 321

Day 321: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 322

Day 322: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 323

Day 323: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 324

Day 324: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 325

Day 325: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 326

Day 326: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 327

Day 327: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 328

Day 328: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 329

Day 329: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 330

Day 330: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 331

Day 331: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 332

Day 332: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 333

Day 333: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 334

Day 334: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 335

Day 335: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)

## Day 336

Day 336: Write a program that prints numbers from 1 to 100 using a loop. (Modify or expand this task for more complexity.)

## Day 337

Day 337: Count the number of vowels in a string input by the user. (Modify or expand this task for more complexity.)

## Day 338

Day 338: Check if a given word is a palindrome. (Modify or expand this task for more complexity.)

## Day 339

Day 339: Store names in an array and search for a specific name. (Modify or expand this task for more complexity.)

## Day 340

Day 340: Display the multiplication table of a number entered by the user. (Modify or expand this task for more complexity.)

## Day 341

Day 341: Sum all elements in an integer array. (Modify or expand this task for more complexity.)

## Day 342

Day 342: Find the maximum and minimum values in an array. (Modify or expand this task for more complexity.)

## Day 343

Day 343: Sort an array in ascending order. (Modify or expand this task for more complexity.)

## Day 344

Day 344: Find duplicate elements in an array. (Modify or expand this task for more complexity.)

## Day 345

Day 345: Use a 2D array to store student names and grades. (Modify or expand this task for more complexity.)

## Day 346

Day 346: Compute the average grade of 5 students using a 2D array. (Modify or expand this task for more complexity.)

## Day 347

Day 347: Create a menu system to add, display, and search records. (Modify or expand this task for more complexity.)

## Day 348

Day 348: Count the frequency of each character in a string. (Modify or expand this task for more complexity.)

## Day 349

Day 349: Reverse a string entered by the user. (Modify or expand this task for more complexity.)

## Day 350

Day 350: Simulate a login system with stored usernames and passwords. (Modify or expand this task for more complexity.)

## Day 351

Day 351: Implement a class called 'Employee' with basic fields. (Modify or expand this task for more complexity.)

## Day 352

Day 352: Use inheritance to create a subclass 'Manager' from 'Employee'. (Modify or expand this task for more complexity.)

## Day 353

Day 353: Read and write data to a text file. (Modify or expand this task for more complexity.)

## Day 354

Day 354: Handle exceptions like division by zero and invalid input. (Modify or expand this task for more complexity.)

## Day 355

Day 355: Generate and store random numbers in a list. (Modify or expand this task for more complexity.)

## Day 356

Day 356: Use a Dictionary to store and retrieve values by key. (Modify or expand this task for more complexity.)

## Day 357

Day 357: Calculate the factorial of a number using recursion. (Modify or expand this task for more complexity.)

## Day 358

Day 358: Simulate a bank system with deposit and withdraw functionality. (Modify or expand this task for more complexity.)

## Day 359

Day 359: Create a grading system that assigns letter grades based on average. (Modify or expand this task for more complexity.)

## Day 360

Day 360: Display a calendar for the current month using DateTime. (Modify or expand this task for more complexity.)

## Day 361

Day 361: Create a C# program that displays 'Hello, World!'. (Modify or expand this task for more complexity.)

## Day 362

Day 362: Write a program that asks the user to input their name and then greets them by name. (Modify or expand this task for more complexity.)

## Day 363

Day 363: Create a program that checks whether a number is even or odd. (Modify or expand this task for more complexity.)

## Day 364

Day 364: Write a program that determines the largest of three user-input numbers. (Modify or expand this task for more complexity.)

## Day 365

Day 365: Create a simple calculator that performs basic arithmetic operations. (Modify or expand this task for more complexity.)