**Topic: User Defined Data Type**

**Collaboration Homework #1:**  Calculate a person’s Age: Age user defined data type.

**Points: 30**

**Due date**: January 29, 2021. 2 bonus points if submitted by January 27.

Age is a derived attribute. Rather than storing a person’s age which changes every year, it is more practical to store a person’s birthdate and calculate the age based on current date and birthdate.

**Homework Rubric.**

**3 = Good     2 = Average     1 = Below Average     0 = Not gradable**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criteria** | **Comments** | **Points** |  |  |  |
| **1.**The program fully implements the solution to the given problem |  | 3 | 2 | 1 | 0 |
| **1.**The solution produces the correct result |  | 3 | 2 | 1 | 0 |
| **2.**The program runs without error, has reasonable error checking and exception handling |  | 3 | 2 | 1 | 0 |
| **3.**The user is prompted for input when required |  | 3 | 2 | 1 | 0 |
| **4.**The user is presented with appropriate feedback as to the results of an operation |  | 3 | 2 | 1 | 0 |
| **5.**All project files and source code required to build and execute are submitted |  | 3 | 2 | 1 | 0 |
| **6.**Code is properly indented, and adequately commented |  | 3 | 2 | 1 | 0 |
| **7.**Code is well written and free from unnecessary complexity or redundancy |  | 3 | 2 | 1 | 0 |
| **8.**Programmer defined symbol names are reasonable (variable names, class names, function names, etc...) |  | 3 | 2 | 1 | 0 |
| **9.**The user interface is accurate and is free from misspelled words and bad grammar |  | 3 | 2 | 1 | 0 |
| **Total Points out of 30:** |  |  |  |  |  |
| **Comments:** |  |  |  |  |  |

**What to submit:**

You need to **submit the following to blackboard**:

1. Project **source code** in .cpp file or other high-level language that is ready to run.
2. **Screenshots of at least 5 different sample input/output**, one of the input/output combinations must be for an invalid input value for one of the age parameters (year, month or day)

**Tasks to be completed:**

Create an abstract data type age. One of the methods will calculate the age of a person. The data for the code can be stored as either structures or classes. This code can be written as a console application in C++ or similar language to meet the following requirements.

1. The program should accept the users input and check for valid/invalid input and warn then the user of any invalid:

Sample input:

What is the First Name: Joe

What is the Last Name: Watson

What is the birth year: 2004

What is the birth month: 7

What is the birth day: 4

1. Use system time to identify today’s date
2. The output should include the person’s name, date of birth, today’s date being used and the person’s age:

**Sample output**:

Joe Watson, age 14 was born on 7-4- 2004 and today is 1-30-2019

1. Use a separate file to store the data as structures (or equivalent constructs- e.g., classes would also be a good choice) to store the person’s information.

The code below is not complete: refer to it if needed. Make sure to organize the code into modules (e.g., methods) also when writing the code, you need to take into account the month and the day. Add validation, and comments to the code.

#include <iostream>

using namespace std;

struct perosn

{

char name[20];

int day;

int month;

int year;

..

}person1;

int main()

{

..

}

**Scenario**

1. Ask for name and the year, month, and day of birth
   1. Make sure the entries are valid
2. Find current year, month and day
3. Find age = current date- person’s birth date
4. Display the result