ICT 283

Assignment 1

Lim Wen Chao CT0360379/34368872

Table of Contents

[UML 1](#_Toc97172978)

[Data Dictionary 2](#_Toc97172979)

[Algorithm 2](#_Toc97172980)

[High Level 2](#_Toc97172981)

[Low Level 3](#_Toc97172982)

[Test plan 3](#_Toc97172983)

[Date Unit Test 3](#_Toc97172984)

[Time Unit Test 3](#_Toc97172985)

[Weather Unit Test 4](#_Toc97172986)

[Vector Unit Test 4](#_Toc97172987)

[LogicHelper Unit Test 6](#_Toc97172988)

[ValidationHelper Unit Test 7](#_Toc97172989)

[Main SIT (Using Test.csv) 8](#_Toc97172990)

[Output of test run(s) 10](#_Toc97172991)

[Date Unit Test 10](#_Toc97172992)

[Time Unit Test 11](#_Toc97172993)

[Weather Unit Test 11](#_Toc97172994)

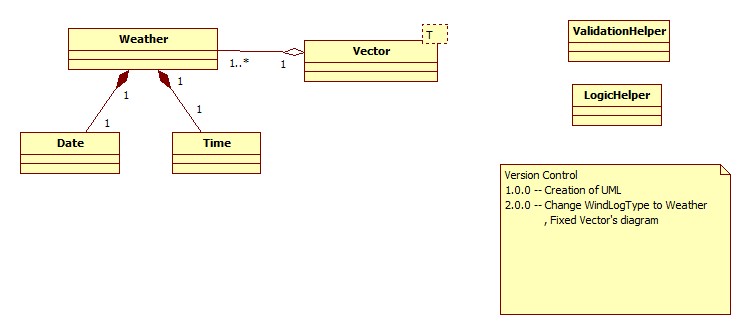
[Vector Unit Test 12](#_Toc97172995)

[LogicHelper Unit Test 13](#_Toc97172996)

[ValidationHelper Unit Test 15](#_Toc97172997)

[Main SIT 16](#_Toc97172998)

# UML



High-Level UML

Diagram

Description automatically generated with medium confidence

Low-level UML

# Data Dictionary



# Algorithm

## High Level

1. Get data file
2. Read data file line by line
   1. Split columns
   2. Get Datetime (WAST)
      1. Split datetime into date and time
      2. Store date in Date class
      3. Store time in Time class
   3. Get Wind speed (S)
   4. Get Solar radiation (SR)
   5. Get Ambient air temperature (T)
3. Store extracted data into custom data type WindLogType
4. Store WindLogType into Vector
5. Repeat Step 2-4 till end of file
6. Display menu
7. If option 1 is chosen
   1. Ask user for month and year input
   2. Search Vector for WindLogType with same month and year in date
   3. Sum all the wind speed and air temperature and count the number of WindLogType that matches
   4. Divide wind speed and air temperature by count to get average
   5. Convert wind speed from m/s to km/h
   6. Prints result
8. If option 2 is chosen
   1. Ask user for year input
   2. Search Vector for WindLogType with same year in date
   3. Sum all the wind speed and air temperature and count the number of WindLogType that matches
   4. Divide wind speed and air temperature by count to get average
   5. Convert wind speed from m/s to km/h
   6. Prints result
   7. Loop b - f for month 1-12
9. If option 3 is chosen
   1. Ask user for year input
   2. Search vector for WindLogType with same year and month is 1 in date
   3. Sum all solar radiation
   4. Convert W/m^2 to kWh/m^2
   5. Print result
   6. Repeat step b – d for month 2-12
10. If option 4 is chosen
    1. Ask user for year input
    2. Search vector for WindLogType with same year and month is 1 in date
    3. Get Average wind speed, Average air temperature and total solar radiation (use functions from made for option 2 and 3 for this)
    4. Print result
    5. Repeat step b – d for month 2-12
11. If option 5 is chosen
    1. Exit program

## Low Level



# Test plan

## Date Unit Test

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check constructor correctly initializes the data and all getters are able to retrieve the data | Default Constructor  Day: -1  Month: -1  Year: -1  Constructor  Day: 1  Month: 1  Year: 1 | Pass |
| 2 | Check that setters are working correctly | Test Setters  Day: 3  Month: 3  Year: 3 | Pass |

## Time Unit Test

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check constructor correctly initializes the data and all getters can retrieve the data | Default Constructor  Hour: -1  Minute: -1  Constructor  Hour: 1  Minute: 1 | Pass |
| 2 | Check that setters are working correctly | Test Setters  Hour: 3  Minute: 3 | Pass |

## Weather Unit Test

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check that constructor correctly initialize the data and all getters can retrieve the data | Check constructor  Date: -1/-1/-1  Time: 0-1:0-1  Wind Speed: -1  Solar Radiation-1  Air Temperature: -1 | Pass |
| 2 | Check that setters are working correctly | Test Setters  Date: 16/3/2016  Time: 09:15  Wind Speed: 23.1  Solar Radiation53.15  Air Temperature: 53.25 | Pass |

## Vector Unit Test

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check that Constructor can correctly initialize an empty Vector and Operator[] will throw out of range exception | Check Constructor  Try retrieve: Unable to retrieve any elements | Pass |
| 2 | Check that the copy constructor can correctly deep copy another Vector | Test Copy Constructor  Retrieving test Vector: testpointer: <pointer>  Retrieving test Vector: test2pointer: <pointer>  Retrieving test Vector: test3pointer: <pointer>  Retrieving copy Vector: testpointer: <pointer>  Retrieving copy Vector: test2pointer: <pointer>  Retrieving copy Vector: test3pointer: <pointer>  \*ensure pointer values are all different | Pass |
| 3 | Check that Size function can correctly return the size of the vector | Check Size of test vector: 3 | Pass |
| 4 | Check that Capacity function can correctly return the capacity of the vector | Check Capacity of test Vector: 4 | Pass |
| 5 | Check that the deconstructor can successfully delete the Vector | Test deconstructor  Size of copy Vector before decontructing: 3  Size of copy Vector after decontructing: 0 | Pass |
| 6 | Check that Operator= can correctly deep copy another Vector | Test Operator=  Retrieve test Vector pointer: <pointer>  Retrieving test Vector[0]: test, pointer: <pointer>  Retrieving test Vector[1]: test2, pointer: <pointer>  Retrieving test Vector[2]: test3, pointer: <pointer>  Retrieve copy Vector pointer: <pointer>  Retrieving copy Vector[0]: test, pointer: <pointer>  Retrieving copy Vector[1]: test2, pointer: <pointer>  Retrieving copy Vector[2]: test3, pointer: <pointer>  \*Ensure all pointers are different | Pass |
| 7 | Check that Resize function can correctly increase the capacity of the vector by multiple of 2 if it isn’t enough | Check Resize(int) function  current size and capasity of vector, size(3) capasity(4)  size and capasity of vector after resizing to 5, size(5) capasity(6)  size and capasity of vector after resizing to 3, size(3) capasity(6) |  |
| 8 | Check that Erase function can correctly remove an element in a vector while shifting all elements after the index to the left | Check Erase(int)  Retrieving Vector index 0: test  Retrieving Vector index 1: test2  Retrieving Vector index 2: test3  After erasing element at index 1  Retrieving Vector index 0: test  Retrieving Vector index 1: test3 | Pass |
| 9 | Check that Insert function can correctly insert an element in a vector while shifting all elements from the index to the right | Check Insert(int,<T>)  Retrieving copy Vector index 0: test  Retrieving copy Vector index 1: test3  Inserted new element at index 1  Retrieving Vector index 0: test  Retrieving Vector index 1: Test  Retrieving Vector index 2: test3 | Pass |
| 10 | Check that clear function can correctly remove all elements in a vector | Check clear()  Size of Vector before clearing: 3  Size of Vector after clearing: 0 | Pass |

## LogicHelper Unit Test

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check that RemoveLeadingZeros function can correctly remove the leading zeros when given a string | Test RemoveLeadingZeros(string)  Test RemoveLeadingZeros(08): 8  Test RemoveLeadingZeros(08.9): 8.9 | Pass |
| 2 | Check that DisplayMenu function can successfully display the menu | Test DisplayMenu()  1. The average wind speed and average ambient air temperature for a specified month and year.  2. Average wind speed and average ambient air temperature for each month of a specified year.  3. Total solar radiation in kWh/m2 for each month of a specified year.  4. Write average wind speed (km/h), average ambient air temperature and total solar radiation in kWh/m2 for each month of a specified year to CSV.  5. Exit the program. | Pass |
| 3 | Check that IntMonthToString function can correctly return the month in string when provided with a month in integer | Test IntMonthToString(unsigned int)  Test IntMonthToString(8): August  Test IntMonthToString(13):  Error converting int month to string  ERROR | Pass |
| 4 | Check that ConvertWindMStoKMH function can correctly convert windspeed provided in m/s to km/h | Test ConvertWindMStoKMH(float)  Test ConvertWindMStoKMH(10): 36  Test ConvertWindMStoKMH(10.5): 37.8 | Pass |
| 5 | Check that ConvertSolarRadiationWMtoKWH function can correctly convert solar radiation provided in W/m2 to kWh/m2 | Test ConvertSolarRadiationWMtoKWH(float)  Test ConvertSolarRadiationWMtoKWH(120): 0.02  Test ConvertSolarRadiationWMtoKWH(120.42): 0.02007 | Pass |
| 6 | Check that LoadDataFileCSV function can correctly load csv data file | Test LoadDataFileCSV function  31/3/2016,09:00,6,512,20.74  31/3/2016,09:10,5,565,20.97  31/3/2016,09:20,5,574,20.92  31/3/2016,09:30,5,623,21.63  31/3/2016,09:40,6,617,22.39 | Pass |
| 7 | Check that AverageWindSpeedMonth function can correctly find and calculate the average windspeed for a month | Test AverageWindSpeedMonth(vector,int,int)  Test AverageWindSpeedMonth(TestLog,3,2016): 5.4  Test AverageWindSpeedMonth(TestLog,1,2020): -1 | Pass |
| 8 | Check that AverageAirTemperatureMonth function can correctly find and calculate the average ambient air temperature for a month | Test AverageAirTemperatureMonth(vector,int,int)  Test AverageAirTemperatureMonth(TestLog,3,2016): 21.33  Test AverageAirTemperatureMonth(TestLog,1,2020): -1 | Pass |
| 9 | Check that SumSolarRadiationMonth function can correctly find and calculate the total solar radiation for a month | Test SumSolarRadiationMonth(vector,int,int)  Test SumSolarRadiationMonth(TestLog,3,2016): 2891  Test SumSolarRadiationMonth(TestLog,1,2020): -1 | Pass |
| 10 | Check that WriteAppendFile able to correctly write to "WindTempSolar.csv" | Test WriteAppendFile(string)  \*Look for the WindTempSolar.csv file  teststring  teststring2 | Pass |

## ValidationHelper Unit Test

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check that IsValidDate function can correctly determine if a date is valid when given the day, month and year | Checking IsValidDate(int day, int month, int year)  Positive case (29/2/2020): 1  Negative case, wrong day (31/2/2020): 0  Negative case, wrong month (29/13/2020): 0  Negative case, wrong year (29/2/99): 0  Negative case, not leap year (29/2/2022): 0 | Pass |
| 2 | Check that IsValidDate function can correctly determine if a date is valid when given the month and year only | Checking IsValidDate(int month, int year)  Positive case (2/2020): 1  Negative case, wrong month (13/2020): 0  Negative case, wrong year (2/99): 0 | Pass |
| 3 | Check that IsValidMonth function can correctly determine if a int is valid month | Checking IsValidMonth(int month)  Positive case (2): 1  Negative case, wrong month (13): 0 | Pass |
| 4 | Check that IsLeapYear function can correctly determine if a year is a leap year | Checking IsLeapYear(int year)  Positive case (2020): 1  Negative case (2022): 0 | Pass |
| 5 | Check that IsValid24HourTime function can correctly determine if a given hour and minute is valid 24 hour time | Checking IsValid24HourTime(int hour, int minute)  Positive case (8,50): 1  Negative case, wrong hour (25,50): 0  Negative case, wrong minute (24,60): 0 | Pass |
| 6 | Check that IsNumber function can correctly determine if a provided string is positive int/float | Checking IsNumber(string input)  Positive case (415): 1  Positive case (63.41): 1  Negative case, negative int (-1): 0  Negative case, negative float (-1.2): 0  Negative case, too many decimal points (123456.123456789012345): 1  Negative case, octal numbers (08): 0 | Pass |
| 7 | Check that IsInteger function can correctly determine if a provided string is a positive integer | Checking IsInteger(string input)  Positive case (415): 1  Negative case (45.21): 0  Negative case (-1): 0  Negative case (08): 0 | Pass |

## Main SIT (Using Test.csv)

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Description | Expected Output | Passed |
| 1 | Check that if user entered a invalid file name the program will repeatedly prompt for file name until a file is found and able to be opened | File not found, please enter a existing file's name  Enter file name to read (Must be in data folder) | Pass |
| 2 | Check that invalid month or year input while using function 1, it will result in error | Error: invalid month. Expects integer. | Pass |
| 3 | Check that invalid month and year combination will result in error and stopping of program | Error : invalid date. | Pass |
| 4 | Check that function 1 can print correctly average wind speed and average air temperature when data is found for the specific month and year in km/h and degrees C respectively | March 2016: 19.44 km/h, 21.33 degrees C | Pass |
| 5 | Check that function 1 will show no data if no data is found for the specific month and year | January 2016: No Data | Pass |
| 6 | Check that if user entered invalid year while using function 2, it will result in error | Error: invalid year. Expects integer. | Pass |
| 7 | Check that function 2 can correctly print the average wind speed and average air temperature of all 12 months in the specific year in km/h and degrees C respectively and no data if no data is found for the month | 2016  January 2016: No Data  February 2016: No Data  March 2016: 19.44 km/h, 21.33 degrees C  April 2016: No Data  May 2016: No Data  June 2016: No Data  July 2016: No Data  August 2016: No Data  September 2016: No Data  October 2016: No Data  November 2016: No Data  December 2016: No Data | Pass |
| 8 | Check that if user entered invalid year while using function 3, it will result in error | Error: invalid year. Expects integer. | Pass |
| 9 | Check that function 3 can correctly print the total solar radiation for each month for a specific year in kWh/m2 and no data if no data is found for the month | 2016  January 2016: No Data  February 2016: No Data  March 2016: 0.481833 kWh/m²  April 2016: No Data  May 2016: No Data  June 2016: No Data  July 2016: No Data  August 2016: No Data  September 2016: No Data  October 2016: No Data  November 2016: No Data  December 2016: No Data | Pass |
| 10 | Check that if user entered invalid year while using function 4, it will result in error and stopping of program | Error: invalid year. Expects integer. | Pass |
| 11 | Check that function 4 can correctly output the average wind speed, average temperature and total solar radiation for each month of a specific year in km/h, degrees C and kWh/m2 and no line for months with no data | \*Check WindTempSolar.csv  2016  March,19.440001,21.329998,0.481833 | Pass |
| 12 | Check that function 5 can quit the program | \*Program exits | Pass |
| 13 | Check that inputting any function other than 1-5 will return in error and repeatedly display the menu | Error: Unknown command. Only numbers 1-5 accepted.  1. The average wind speed and average ambient air temperature for a specified month and year.  2. Average wind speed and average ambient air temperature  for each month of a specified year.  3. Total solar radiation in kWh/m2 for each month of a specified year.  4. Write average wind speed (km/h), average ambient air temperature and total solar radiation in kWh/m2 for each month of a specified year to CSV.  5. Exit the program. | Pass |

# Output of test run(s)

## Date Unit Test



Text

Description automatically generated



Graphical user interface, text

Description automatically generated

## Time Unit Test



Text

Description automatically generated



Graphical user interface, text, application

Description automatically generated

## Weather Unit Test

Text

Description automatically generated



Text

Description automatically generated

## Vector Unit Test







Text

Description automatically generated











Graphical user interface, text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated



Graphical user interface, text

Description automatically generated

## LogicHelper Unit Test



A black background with white text

Description automatically generated with medium confidence



Text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated with low confidence



Text

Description automatically generated



Text

Description automatically generated







Text

Description automatically generated with medium confidence





Text

Description automatically generated

## ValidationHelper Unit Test



Text

Description automatically generated



Text

Description automatically generated



A black background with white text

Description automatically generated with low confidence



Text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated

## Main SIT



Graphical user interface, text

Description automatically generated



Text

Description automatically generated



Text

Description automatically generated















Text

Description automatically generated







Text

Description automatically generated







Graphical user interface, text

Description automatically generated



Program exits



Text

Description automatically generated

# Evaluation

