## Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| Date | Class |  | Represents a Date Class. |
| m\_day | unsigned | - | An unsigned day of a date. |
| m\_month | unsigned | - | An unsigned month of a date. |
| m\_year | unsigned | - | An unsigned year of a date. |
| Date() | procedure | + | Sets the day, month, and year to 0. |
| Date(day, month, year) | procedure | + | Sets a specific day, month, and year. |
| GetDay() | query | + | Gets the day of a date. |
| GetMonth() | query | + | Gets the month of a date. |
| GetYear() | query | + | Gets the year of a date. |
| SetDay(unsigned day) | procedure | + | Sets the day of a date. |
| SetMonth(unsigned month) | procedure | + | Sets the month of a date. |
| SetYear(unsigned year) | procedure | + | Sets the year of a date. |
| operator == ( const Date &otherDate ) | Boolean | + | checks whether two Date objects have the same day, month, and year |
| operator != ( const Date &otherDate ) | Boolean | + | checks whether this Date objects is not having the same day, month, and year |
| operator < ( const Date &otherDate ) | Boolean | + | checks whether this Date objects is lesser than the other Date Object. |
| operator > ( const Date &otherDate ) | Boolean | + | checks whether this Date objects is greater than the other Date Object. |
| &operator << (ostream, date) | query | + | Format Date object, inserts into output stream. |
| &operator >> (istream, date) | query | + | Reads a Date Object from input stream. |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| Time | Class |  | Represents a Time Class. |
| m\_hour | unsigned | - | An unsigned hour of a time. |
| m\_minute | unsigned | - | An unsigned minute of a time. |
| Time() | procedure | + | Sets the hour, and minute to 0. |
| Time(hour, minute) | procedure | + | Sets a specific hour, and minute. |
| GetHour() | query | + | Gets the hour of a time. |
| GetMinute() | query | + | Gets the minute of a time. |
| SetHour(unsigned hour) | procedure | + | Sets the hour of a time. |
| SetMinute(unsigned minute) | procedure | + | Sets the minute of a time. |
| operator == ( const Time &otherTime ) | Boolean | + | checks whether two Time objects have the same hour, and minute. |
| operator != ( const Time &otherTime ) | Boolean | + | checks whether this Time objects is not having the same hour, and minute. |
| operator < ( const Time & otherTime) | Boolean | + | checks whether this Time objects is lesser than the other Time Object. |
| operator > ( const Time & otherTime) | Boolean | + | checks whether this Time objects is greater than the other Time Object. |
| &operator << (ostream, time) | query | + | Format Time object, inserts into output stream. |
| &operator >> (istream, time) | query | + | Reads a Time Object from input stream. |

* Below Measurement class as the base class (Parent class)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| Measurement | Class |  | Represents an Observations Class. |
| m\_measurement | float | - | A float measurement of an observation. |
| Measurement() | procedure | + | Sets the measurement to 0.0 |
| Measurement(measurement) | procedure | + | Sets a specific measurement. |
| GetMeasurement() | query | + | Gets the measurement of an observation. |
| SetMeasurement(measurement) | procedure | + | Sets the measurement of an observation. |

* Below an inheritance WindSpeed Class from Measurement class (Child class)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| WindSpeed | Class |  | Represents a WindSpeed Class. |
| WindSpeed() | procedure | + | Sets the wind speed measurement to 0.0 |
| WindSpeed(measurement) | procedure | + | Sets a specific wind speed measurement. |
| ConvertUnit( &measurement) | float | + | Returns the converted unit for kWh/m2 |
| ~~&operator << (ostream, W)~~ | ~~query~~ | ~~+~~ | ~~Format WindSpeed object, inserts into output stream.~~ |
| ~~&operator >> (istream, W)~~ | ~~query~~ | ~~+~~ | ~~Reads a WindSpeed Object from input stream.~~ |

* Below an inheritance SolarRadiation Class from Measurement class (Child class)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| SolarRadiation | Class |  | Represents a SolarRadiation Class. |
| SolarRadiation () | procedure | + | Sets the solar radiation measurement to 0.0 |
| SolarRadiation(measurement) | procedure | + | Sets a specific solar radiation measurement. |
| ConvertUnit( &measurement) | float | + | Returns the converted unit for the reading |
| ~~&operator << (ostream, SR)~~ | ~~query~~ | ~~+~~ | ~~Format SolarRadiation object, inserts into output stream.~~ |
| ~~&operator >> (istream, SR)~~ | ~~query~~ | ~~+~~ | ~~Reads a SolarRadiation Object from input stream.~~ |

* Below an inheritance Temperature Class from Measurement class (Child class)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| Temperature | Class |  | Represents a Temperature Class. |
| Temperature () | procedure | + | Sets the temperature measurement to 0.0 |
| Temperature(measurement) | procedure | + | Sets a specific temperature measurement. |
| ~~&operator << (ostream, TEMP)~~ | ~~query~~ | ~~+~~ | ~~Format Temperature object, inserts into output stream.~~ |
| ~~&operator >> (istream, TEMP)~~ | ~~query~~ | ~~+~~ | ~~Reads a Temperature Object from input stream.~~ |

* Below composite class of Date, Time, WindSpeed, SolarRadiation, Temperature

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| SensorRecType | Class |  | Represents a SensorRecType Class. |
| m\_date | Date | - | A Date of a date Type. |
| m\_time | Time | - | A Time of a time Type. |
| m\_windSpeed | WindSpeed | - | A WindSpeed of a SensorRecType. |
| m\_solarRadiation | SolarRadiation | - | A SolarRadiation of a SensorRecType. |
| m\_temperature | Temperature | - | A Temperature of a SensorRecType. |
| SensorRecType() | procedure | + | Sets the date, time, windspeed, solar radiation and temperature to default value. |
| SensorRecType(date, time, windspeed, solarRadiation, temperature) | procedure | + | Sets a specific date, time, windspeed, solarRadiation, temperature. |
| GetSensorDate() | query | + | Gets the date of a SensorRecType. |
| GetSensorTime() | query | + | Gets the time of a SensorRecType. |
| GetSensorWindSpeed() | query | + | Gets the wind speed of a SensorRecType. |
| GetSensorSolarRadiation() | query | + | Gets the solar radiation of a SensorRecType. |
| GetSensorTemperature() | query | + | Gets the temperature of a SensorRecType. |
| SetSensorDate (Date date) | procedure | + | Sets the date of a SensorRecType. |
| SetSensorTime (Time time) | procedure | + | Sets the time of a SensorRecType. |
| SetSensorWindSpeed(  WindSpeed windspeed) | procedure | + | Sets the wind speed of a SensorRecType. |
| SetSensorSolarRadiation(  SolarRadiation solarRadiation) | procedure | + | Sets the solar radiation of a SensorRecType. |
| SetSensorTemperature(  Temperature temperature) | procedure | + | Sets the temperature of a SensorRecType. |
| &operator << (ostream, SRT) | query | + | Format SolarRecType object, inserts into output stream. |
| &operator >> (istream, SRT) | query | + | Reads a SolarRecType Object from input stream. |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| Vector | Class<T> |  | Represents a Vector Class. |
| INITIAL\_CAPACITY | unsigned | - | Represents the initial capacity of the vector. |
| INITIAL\_VALUE | unsigned | - | Represents the default initial value. |
| m\_capacity | unsigned | - | Represents the current capacity of the vector. |
| m\_used | unsigned | - | Represents the number of elements used in the vector. |
| m\_array | T\* | - | Represents the pointer to the dynamic array storing vector elements. |
| Vector() | Constructor | + | Default constructor. Initializes the vector with an initial capacity of two and creates a new dynamic array on the heap memory. |
| ~Vector() | Destructor | + | Destructor. Frees the dynamically allocated memory for the vector, sets the capacity and used elements to zero. |
| Vector(const Vector &otherVector) | Copy constructor | + | Copy constructor. Creates a new vector as a copy of another vector. |
| operator=(const Vector &otherVector) | Assignment operator | + | Assignment operator. Assigns the contents of another vector to this vector. |
| GetCapacity() | unsigned | + | Get the current capacity of the vector. |
| GetUsed() | unsigned | + | Get the number of elements used in the vector. |
| IsEmpty() | bool | + | Check if the vector is empty. |
| Start() | T\* | + | Pointer that points to the start of the vector index [0]. |
| Finish() | T\* | + | Pointer that points to the end of the vector index. |
| operator[](const unsigned &index) | T& | + | Access an element by vector's index. |
| operator[](const unsigned &index) const | const T& | + | Const version of access an element by vector's index. |
| InsertLast(const T &object) | bool | + | Insert an object to the end of the vector. |
| InsertAt(const T &object, const unsigned &index) | bool | + | Insert an object to a desired index of the vector. |
| RemoveLast() | bool | + | Remove an object from the end index of the vector. |
| RemoveAt(const unsigned &index) | bool | + | Remove an object from a desired index of the vector. |
| Deallocate(T \*&ptr) | void | - | Deallocate memory for a given pointer. |
| CopyVector(const Vector &otherVector) | void | - | Copy the contents of another vector. |
| CopyArray(T \*copiedArray, T \*otherArray) | void | - | Copy the contents of one array to another. |
| Resize() | void | - | Resize the vector by doubling its capacity. |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| MapAov | Class |  | Represents a MapAov Class. |
| NUM\_MTH | const unsigned | + | Number of months in a year. |
| m\_data | map<unsigned, array<Vector<SensorRecType>, NUM\_MTH>> | - | Map storing arrays of sensor data vectors organized by month and year. |
| GetSensorData() | const map<unsigned, array<Vector<SensorRecType>, NUM\_MTH>>& | + | Get a constant reference to the sensor data map. |
| GetSensorData() | map<unsigned, array<Vector<SensorRecType>, NUM\_MTH>>& | + | Get a reference to the sensor data map. |
| InsertSensorData(const SensorRecType &sensorData) | void | + | Insert sensor data into the map consisting of arrays of Vectors. |
| SumOfMeasurementSwitch(const Vector<SensorRecType> &sensorRecords, SensorMeasurementType sensorMeasurementType) | float | + | Switch function to compute the sum of different types of measurements. |
| MeanOfMeasurementSwitch(const Vector<SensorRecType> &sensorRecords, SensorMeasurementType sensorMeasurementType) | float | + | Switch function to compute the mean of different types of measurements. |
| SampleStandardDeviationMeasurementSwitch(const Vector<SensorRecType> &sensorRecords, SensorMeasurementType sensorMeasurementType) | float | + | Switch function to compute the sample standard deviation of different types of measurements. |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| Processor | Class |  | Represents a Processor Class. |
| s\_instance | Processor\* | - | Pointer to the singleton instance of the Processor class. |
| m\_dataRead | unsigned | - | Counter for the number of data read. |
| m\_dataInserted | unsigned | - | Counter for the number of data inserted. |
| Processor() | Constructor | - | Private constructor to prevent instantiation of Processor class objects. |
| GetInstance() | Processor& | + | Returns the singleton instance of Processor. |
| LoadCSVFilePathToVector(Vector<string> &csvFilePath, ifstream &input, const string &filename) | void | + | Loads the CSV file paths from a text file into a vector. |
| LoadCSVData(Vector<string> &csvFilePath, MapAov &mapAov) | void | + | Loads CSV data from a vector of file paths into a MapAov object. |
| OutputStreamMeasurement(SensorRecType &sensorRecType, ofstream &outputFile, unsigned &month, float &speedMean, float &speedSsd, float &tempMean, float &tempSsd, float &srSum) | bool | + | Records sensor data to an output file and computes statistical measurements. |
| LoadDisplayCount() | void | - | Loads the display count for the Processor instance. |
| PopulateData(recordVec, records, path\_name) | Boolean | + | Populate wind record data from a specific file to the vector. |

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Protection | Description |
| InputValidator | Class |  | Represents a InputValidator Class. |
| m\_userYear | unsigned | - | Holds the user input for the year. |
| m\_userMonth | unsigned | - | Holds the user input for the month. |
| InputValidator() | Constructor | + | Default constructor for the InputValidator class. |
| MonthValidation(string &month) | unsigned | + | Validates user input for the month. Returns the validated user input for the month. |
| YearValidation(string &year) | unsigned | + | Validates user input for the year. Returns the validated user input for the year. |
| IsMonthInputValid() | bool | - | Checks if the user input for the month is valid. Returns true if the input is invalid, false otherwise. |
| IsYearInputValid() | bool | - | Checks if the user input for the year is valid. Returns true if the input is invalid, false otherwise. |

## Test Plan

Date Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | day: 0, month:0, year: 0 | Y | Default constructor should initialize day, month, and year to the initial value. |
| 2 | Constructed - Valid input parameters | day: 11, month: 12, year: 2002 | day: 11, month: 12, year: 2002 | Y | The constructor should correctly initialize day, month, and year with the provided value. |
| 3 | Getter - GetDay | Set day to 15 | GetDay() returns 15 | Y | The GetDay() function should return the day set previously. |
| 4 | Getter - GetMonth | Set month to 7 | GetMonth() returns 7 | Y | The GetMonth() function should return the month set previously. |
| 5 | Getter - GetYear | Set year to 2022 | GetYear() return 2022 | Y | The GetYear() function should return the year set previously. |
| 6 | Setter - SetDay | Set day to 10 | GetDay() returns 10 | Y | The SetDay() function should correctly set the day. |
| 7 | Setter - SetMonth | Set month to 3 | GetMonth() returns 3 | Y | The SetMonth() function should correctly set the month. |
| 8 | Setter - SetYear | Set month to 2024 | GetYear() returns 2024 | Y | The SetYear() function should correctly set the year. |
| 9 | operator == Overloading | thisDate: 25/12/2013, otherDate: 25/12/2013 | true | Y | thisDate == otherDate |
| 10 | operator != Overloading | thisDate: 25/12/2013, otherDate: 20/12/2013 | true | Y | thisDate != otherDate |
| 11 | operator < Overloading | thisDate: 20/12/2013, otherDate: 25/12/2013 | true | Y | thisDate < otherDate |
| 12 | operator > Overloading | thisDate: 25/12/2013, otherDate: 20/12/2013 | true | Y | thisDate > otherDate |
| 13 | GetMonthInStr - Valid month | Input: 4 | Returns "April" | Y | The GetMonthInStr() function should return the name if the month corresponding to the input month number. |
| 14 | GetMonthInStr - Invalid month | Input: 15 | Return "Error" | Y | The GetMonthInStr() function should return "Error" for invalid month numbers. |
| 15 | Operator << Overloading | Date: 12/5/2023 | Output: "12 May 2023" | Y | The overloading << operator should correctly output the date in the format "day month year". |
|  | | | | | |

Time Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | hour: 0, minute:0 | Y | Default constructor should initialize hour, and minute to the initial value. |
| 2 | Constructed - Valid input parameters | hour: 12, minute: 12 | hour: 0, minute:0 | Y | The constructor should correctly initialize hour, and minute with the provided value. |
| 3 | Setter - SetHour | Set hour to 11 | GetHour() returns 11 | Y | The SetHour() function should correctly set the hour. |
| 4 | Setter - SetMinute | Set minute to 11 | GetMinute() returns 11 | Y | The SetMinute() function should correctly set the month. |
| 5 | Getter - GetHour | - | GetHour() returns 11 | Y | The GetHour() function should return the hour set previously. |
| 6 | Getter - GetMinute | - | GetMinute() returns 11 | Y | The GetMinute() function should return the minutes set previously. |
| 7 | operator == Overloading | thisTime: 12:20, otherTime: 12:20 | true | Y | thisTime == otherTime |
| 7 | operator != Overloading | thisTime: 12:20, otherTime: 11:10 | true | Y | thisTime != otherTime |
| 7 | operator < Overloading | thisTime: 11:10, otherTime: 12:20 | true | Y | thisTime < otherTime |
| 7 | operator > Overloading | thisTime: 12:20, otherTime: 11:10 | true | Y | thisTime > otherTime |
|  |  |  |  |  |  |
|  | | | | | |

Measurement Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | measurement: 0 | Y | Default constructor should initialize measurement to the initial value. |
| 2 | Constructed - Valid input parameters | measurement: 12.12 | measurement: 12.12 | Y | The constructor should correctly initialize measurement with the provided value. |
| 3 | Setter - SetMeasurement | measurement: 11.11 | GetMeasurement() returns 11.11 | Y | The SetMeasurement() function should correctly set the measurement. |
| 4 | Getter - GetMeasurement | - | GetMeasurement() returns 11.11 | Y | The GetMeasurement() function should return the measurement set previously. |
| A screenshot of a computer program  Description automatically generated | | | | | |

WindSpeed Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | measurement: 0 | Y | Default constructor should initialize measurement to the initial value. |
| 2 | Constructed - Valid input parameters | measurement: 12.12 | measurement: 12.12 | Y | The constructor should correctly initialize measurement with the provided value. |
| 3 | Setter - SetMeasurement | measurement: 11.11 | GetWindSpeed() returns 11.11 | Y | The SetWindSpeed() function should correctly set the measurement. |
| 4 | Getter - GetMeasurement | - | GetWindSpeed() returns 11.11 | Y | The GetWindSpeed() function should return the measurement set previously. |
| 5 | Converting WindSpeed to kWh/m2 | measurement: 30.0 | 108.0 kWh/m2 | Y | Wind Speed converted to kWh/m2 |
|  | | | | | |

Temperature Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | measurement: 0 | Y | Default constructor should initialize measurement to the initial value. |
| 2 | Constructed - Valid input parameters | measurement: 12.12 | measurement: 12.12 | Y | The constructor should correctly initialize measurement with the provided value. |
| 3 | Setter - SetMeasurement | measurement: 11.11 | GetTemperature() returns 11.11 | Y | The SetTemperature() function should correctly set the measurement. |
| 4 | Getter - GetMeasurement | - | GetTemperature() returns 11.11 | Y | The GetTemperature() function should return the measurement set previously. |
| A screenshot of a computer program  Description automatically generated | | | | | |

SolarRadiation Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | measurement: 0 | Y | Default constructor should initialize measurement to the initial value. |
| 2 | Constructed - Valid input parameters | measurement: 12.12 | measurement: 12.12 | Y | The constructor should correctly initialize measurement with the provided value. |
| 3 | Setter - SetMeasurement | measurement: 11.11 | GetSolarRadiation() returns 11.11 | Y | The SetSolarRadiation() function should correctly set the measurement. |
| 4 | Getter - GetMeasurement | - | GetSolarRadiation() returns 11.11 | Y | The GetSolarRadiation() function should return the measurement set previously. |
|  | | | | | |

SensorRecType Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Constructed - Default | None | Sensor data initialized to default values | Y | Default constructor should initialize all sensor data to the default values. |
| 2 | Constructed - Valid input parameters | Date: 24/2/2024, Time:12:30, WindSpeed: 15.5, SolarRadiation: 500, Temperature: 25.5 | Sensor data initialized to the provided values. | Y | The constructor should correctly initialize sensor data fields with the provided value. |
| 3 | Setter - SetMeasurement | Date: 24/2/2024, Time:12:30, WindSpeed: 15.5, SolarRadiation: 500, Temperature: 25.5 | Setters correctly update sensor data. | Y | The SetSensorDate(), SetSensorTime(), SetSensorWindSpeed(), SetSetSensorTemperature(), SetSensorSolarRadiation() function should correctly set the measurement. |
| 4 | Getter - GetMeasurement | - | Getters return correct sensor data. | Y | The GetSensorDate(), GetSensorTime(), GetSensorWindSpeed(), GetSensorTemperature(), GetSensorSolarRadiation() function should return the measurement set previously. |
| A screenshot of a computer program  Description automatically generated | | | | | |

Vector Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Test the initialization of the Vector class. | None | The initial capacity should be set to the default value (2), and the number of used slots should be set to 0. | Y | Default constructor should initialize all sensor data to the default values. |
| 2 | Test on data type initialization of Vector | None | Vector data type set as unsigned data type. | Y | None |
| 3 | Test on insertion of object to Vector's last index. | 1st Object: 10, 2nd Object: 20, 3nd Object: 30, 4th Object: 40 | Objects successfully inserted into Vector's last index. | Y | None |
| 4 | Test on insertion of object of object to Vector's index [1]. | index[0]: 10, index[1]: 20, index[2]: 30, index[3]: 40. Object to insert: 50 at index[1] | Object successfully inserted at specific index. | Y | None |
| 5 | Test on removing object of the last element in Vector. | 10, 20, 30, 40 | Last element removed successfully from the Vector | Y | None |
| 6 | Test on removing object from a specific index | 10, 20, 30, 40; Removing object at index 1 | Object successfully removed from specified index | Y | None |
| 7 | Test for loop output of Vector's element using \*Start And \*Finish | 10, 20, 30, 40 | Pointers correctly traverse Vector | Y | None |
| 8 | Test for index accessor | 10, 20, 30, 40 | Index operator correctly accesses Vector elements | Y | None |
| 9 | Test on Copy Constructor | 10, 20, 30, 40 | Copied Vector matches original Vector | Y | Elements address are to be different on both Vectors. |
| 10 | Test on Equals Assignment Operator | 10, 20, 30, 40 | Assigned Vector matches original Vector | Y | Elements address are to be different on both Vectors. |
| A screenshot of a computer  Description automatically generated | | | | | |

MapAov Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Insert Sensor Data | 10.0, 15.0, 20.0 | Sensor data inserted successful into MapAov object. | Y |  |
| 2 | Sum of measurement switch | 10.0, 15.0, 20.0 | Calculate sum matches expected value | Y |  |
| 3 | Mean of measurement switch | 10.0, 15.0, 20.0 | Calculate mean matches expected value. | Y |  |
| 4 | Sample Standard Deviation measurement switch | 10.0, 20.0 | Calculate standard deviation matches expected value. | Y |  |
| A screenshot of a computer  Description automatically generated | | | | | |

InputValidator Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Description / Condition Tested | Test Data / Values | Expected Output | Passed (Y / N) | Comments |
| 1 | Month Validation Test | Input month: 6 | Expected a range of 1 ~ 12 | Y |  |
| 2 | Year Validation Test | Input year: 6 | Expected a range of 2010 ~ 2030 | Y |  |
| A computer screen with white text  Description automatically generated | | | | | |

## UML ( Low Level UML )

A diagram of a graph

Description automatically generatedA diagram of a computer

Description automatically generated

## UML ( High Level UML )

## A screenshot of a computer Description automatically generatedA screenshot of a computer program Description automatically generated

A screenshot of a computer

Description automatically generated

# Vector Pseudocode

## Deallocate Method

|  |  |  |
| --- | --- | --- |
| Deallocate( &pointer ) | | |
|  | IF array exist which the referenced pointer is pointing at. | |
|  |  | Delete that array which was pointed. |
|  |  | Set the referenced pointer to null. |
|  | END IF | |
| END Deallocate | | |

## Vector Default Constructor

|  |  |
| --- | --- |
| Vector( ) | |
|  | Initialize capacity of 2 to the vector. |
|  | Initialize 0 of used space in the vector. |
|  | Allocate memory for the array. |
| END Vector | |

## Vector Destructor

|  |  |
| --- | --- |
| ~Vector( ) | |
|  | Deallocate the memory pointed to using Deallocate method. |
| END ~Vector | |

## Vector IsEmpty method.

|  |  |
| --- | --- |
| IsEmpty ( ) | |
|  | RETURN true if number of used spaces in the vector is 0. |
| END IsEmpty | |

## Vector Operator [] overload method.

|  |  |
| --- | --- |
| &operator[ ] ( &index ) | |
|  | RETURN the reference to the element at index. |
| END &operator [] | |

## Vector const Operator [] overload method.

|  |  |
| --- | --- |
| const &operator[ ] ( &index ) const | |
|  | RETURN the constant reference to the element at index. |
| END &operator [] | |

## Vector CopyArray method.

|  |  |  |
| --- | --- | --- |
| CopyArray( T type copied array pointer, T type another array pointer ) | | |
|  | FOR looping through | |
|  |  | Copy the elements from the other array to the copied array. |
|  | END FOR | |
| END CopyArray | | |

## Vector Resize method.

|  |  |  |
| --- | --- | --- |
| Resize( ) | | |
|  | Calculate the new capacity for the vector | |
|  | Allocate a new array with the new capacity. | |
|  | IF new array was successfully allocated | |
|  |  | Copy elements from the old array to the new array using the CopyArray method. |
|  | END IF | |
|  | Deallocate the memory of the old array using Deallocate method. | |
|  | Allocate another array with the new capacity | |
|  | IF another array was successfully allocated | |
|  |  | Copy elements from the new array to another array using the CopyArray method. |
|  | END IF | |
|  | Deallocate the memory of the new array using Deallocate method. | |
|  | Update the capacity of the vector. | |
| END Resize | | |

## Vector InsertLast method.

|  |  |  |
| --- | --- | --- |
| InsertLast( constant T &object) | | |
|  | IF the vector is half full | |
|  |  | Resize the vector using the Resize method. |
|  | END IF | |
|  | IF the vector is full | |
|  |  | RETURN false indicating insertion fail. |
|  | END IF | |
|  | Insert the referenced object to the end of the vector. | |
|  | Increment used space of the vector. | |
|  | RETURN true indicating insertion successful. | |
| END InsertLast | | |

## Vector CopyVector method.

|  |  |  |
| --- | --- | --- |
| CopyVector( constant &other vector ) | | |
|  | Copy the capacity of the other vector to the current vector. | |
|  | Copy the number of used spaces of the other vector to the current vector. | |
|  | Allocate a new array with the same capacity as the other vector. | |
|  | IF array was successfully allocated | |
|  |  | Copy the elements from the other vector to the current vector using CopyArray method. |
|  | END IF | |
| END CopyVector | | |

## Vector Copy Constructor method.

|  |  |  |
| --- | --- | --- |
| Vector( constant &other vector ) | | |
|  | IF array in the current is not null | |
|  |  | Deallocate the memory for the array in the current vector. |
|  | END IF | |
|  | Updates the elements from the other vector to the current vector using the CopyVector method. | |
| END CopyVector | | |

## Vector Copy Operator method.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Vector( constant &other vector ) | | | | |
|  | IF the current vector is not the same as the other vector | | | |
|  |  | IF the array in current is not null. | | |
|  |  |  | | Deallocate the memory for the array in the current vector. |
|  |  | END IF | | |
|  |  | Copy the elements from the other vector to the current vector | | |
|  | END IF | |  | |
|  | RETURN a reference to the current vector. | | | |
| END Vector | | | | |

## Vector RemoveLast method.

|  |  |  |
| --- | --- | --- |
| RemoveLast( ) | | |
|  | IF vector is empty | |
|  |  | RETURN false |
|  | END IF | |
|  | Decrement the number of used elements | |
|  | RETURN true | |
| END RemoveLast | | |

# Processor Pseudocode

## Processor GetInstance method.

|  |  |  |
| --- | --- | --- |
| GetInstance( ) | | |
|  | IF static instance is not initialized | |
|  |  | CREATE a new instance |
|  | END IF | |
|  | RETURN the reference to the single instance of Processor | |
| END GetInstance | | |

## Processor LoasCSVFilePathToVector method.

|  |  |  |
| --- | --- | --- |
| LoadCSVFilePathToVector( &csvFilePath, &input, &filename ) | | |
|  | IF input stream is not opened | |
|  |  | DISPLAY error message |
|  | ELSE | |
|  |  | READ file line by line |
|  |  | ADD csv file path to the vector |
|  | END IF ELSE | |
|  | Close the input file | |
| END LoadCSVFilePathToVector | | |

## Processor OutputStreamMeasurement method.

|  |  |
| --- | --- |
| LoadCSVFileData( &sensorRecType, &outputFile, &month, &speedMean, &speedSsd, &tempMean, &tempSsd, &srSum) | |
|  | WRITES the output stream to the output file by referencing the measurements. |
| END LoadCSVFileData | |

# MapAov Pseudocode

## MapAov const GetSensorData method.

|  |  |
| --- | --- |
| const GetSensorData( ) const | |
|  | RETURN the constant reference to the data member variable. |
| END GetInstance | |

## MapAov GetSensorData method.

|  |  |
| --- | --- |
| GetSensorData( ) | |
|  | RETURN the reference to the data member variable. |
| END GetInstance | |

## MapAov InsertSensorData method.

|  |  |  |
| --- | --- | --- |
| InsertSensorData( &sensorData ) | | |
|  | GET the year and month from the sensor data. | |
|  | FIND the entry point for the year in the map data. | |
|  | IF entry does not exist, create it and initialize the array | |
|  |  | CREATE an array of vectors. |
|  |  | INSERT the array into the map |
|  |  | Reacquire the iterator after the insertion. |
|  | END IF | |
|  | ACCESS the vector for the month and add the sensor data(s). | |
| END GetInstance | | |

## MapAov SumOfMeasurementSwitch method.

|  |  |  |  |
| --- | --- | --- | --- |
| SumOfMeasurementSwitch( &sensorRecords, sensorMeasurementType ) const | | | |
|  | SWITCH which measurement type to sum | | |
|  |  | CASE WIND\_SPEED | |
|  |  |  | SUM the retrieved windspeed measurement |
|  |  | BREAK CASE WIND\_SPEED | |
|  |  | CASE AMBIENT\_TEMPERATURE | |
|  |  |  | SUM the retrieved temperature measurement |
|  |  | BREAK CASE AMBIENT\_TEMPERATURE | |
|  |  | CASE SOLAR\_RADIATION | |
|  |  |  | SUM the retrieved solar radiation measurement |
|  |  | BREAK CASE SOLAR\_RADIATION | |
|  | END SWITCH | | |
|  | RETURN the sum | | |
| END SumOfMeasurementSwitch | | | |

## MapAov SampleStandardDeviationMeasurementSwitch method.

|  |  |  |  |
| --- | --- | --- | --- |
| SampleStandardDeviationMeasurementSwitch( &sensorRecords, sensorMeasurementType ) const | | | |
|  | SWITCH which measurement type to sum and sumSquare Each element | | |
|  |  | CASE WIND\_SPEED | |
|  |  |  | SUM the retrieved windspeed measurement and sumSquare each element |
|  |  | BREAK CASE WIND\_SPEED | |
|  |  | CASE AMBIENT\_TEMPERATURE | |
|  |  |  | SUM the retrieved temperature measurement and sumSquare each element |
|  |  | BREAK CASE AMBIENT\_TEMPERATURE | |
|  |  | CASE SOLAR\_RADIATION | |
|  |  |  | SUM the retrieved solar radiation measurement and sumSquare each element |
|  |  | BREAK CASE SOLAR\_RADIATION | |
|  | END SWITCH | | |
|  | Calculate the variance | | |
|  | RETURN the calculated Sample Standard Deviation | | |
| END SampleStandardDeviationMeasurementSwitch | | | |

## MapAov MeanOfMeasurementSwitch method.

|  |  |  |  |
| --- | --- | --- | --- |
| MeanOfMeasurementSwitch( &sensorRecords, sensorMeasurementType ) const | | | |
|  | | Calculate the sum of a measurement | |
|  | | GET the count number of the sensor record; | |
|  | | IF sensor record vector is not empty | |
|  | |  | Calculate the mean (average) |
|  | END IF | | |
| END MeanOfMeasurementSwitch | | | |