|  |  |  |
| --- | --- | --- |
| **Function signature** | | **verify\_shipment***(input)* |
| This function is the main function that runs all validation functions to validate shipment’s data. It returns an array of objects that contain all errors (if exist) associated with the input. The objects in the returned array are arrays that show the errors associated with the input and their type, and where they exist. | | |
| Argument : | * *Input: survey table* | |
| Returns: | Array() [ {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, … ] | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_numberOfShip** *(shipNum,nos,line)* |
| This function validates the number of shipments by checking its presence. The presence check involves checking for shipment number, if it does not exist, it also checks for the entered number of shipments ‘nos’ value and returns the associated error flag. Otherwise, it checks for the allowed characters (it is required to be numeric input). Then, it checks for the shipmentNum value range (the maximum allowed value is 100,000). Next, it checks for shipment number and ‘nos’ values, if any of them is zero and the other is greater than zero it returns an error. It also runs a range value check for shipNum and checks for ‘nos’ value, it returns an error if nos is greater than zero and the shipNum is greater than maxRange. Next, if the shipment number is entered, the function determines the required number of shipments to be reported, then it checks consistency of the required reported number of shipments with the total shipments. The function returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *shipNum: shipment number (string)* * *nos: number of shipments (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |
| **Function signature** | | **MOS\_vs\_ATV***(ATV, MOS, estbWeight, evalres,line)* |
| This function validates the consistency of MOS and ATV values. First, it checks the validity of the values of ATV and MOS. If both values are valid, it finds the difference between the two values ‘dif’ and their ratio ‘ratio’. Next, it runs range value check for ‘dif’ and ‘ratio’ to check their consistency with the entered MOS and ATV values. If no errors are found, it runs a range value check for both ‘dif’ and ‘estbWeight’ to check the consistency of ATV and MOS. The function returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *ATV:* * *MOS:* * *estWeight:* * *evalres:* * *line:* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type:  ​ | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_totShipValue***(totShipVal, totValWeek, ATV, estbWeight, evalres,line)* |
| This function validates the total shipment value with regards the validity of the input and its consistency with the ATV value. It validates the entered value first by checking its presence. If no error is found, it checks for the allowed characters (numeric input is required), it also runs range value check (minRange is 1). It also checks the validity of the entered ATV value, if it is valid, it checks for the consistency of the total shipment value with the ATV value. The function returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *totShipVal: total shipment value* * *totValWeek: total weekly value* * *ATV:* * *estWeight: estimated shipment weight* * *evalres:* * *line: line number* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_moreThan40Ship***(input,line)* |
| This function validates the input if the shipment size is more than 40 or not. It first runs presence check, if input is entered, it runs a look-up linear check to validate the existence of the input in a look-up table (lkup25). The function returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected. | | |
| Argument : | * *Input: yes or no (string)* * *Line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_ship\_ID("yes",1)  Returns: Object { pass: true } | |
| **Function signature** | | **test\_ship\_ID***(input,line)* |
| This function validates the shipment ID by checking its presence. If an input is entered, it checks for allowed characters (alphanumeric input is required). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipment ID (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_ship\_ID("shp123",1)  Returns: Object { pass: true } | |
| **Function signature** | | **test\_ship\_month***(input,line)* |
| This function validates the shipment month by checking its presence. If an input is entered, it checks for allowed characters (numeric input is required). It also checks for the entered value range, this involves checking for the allowed range of values that are allowed to be entered (in this function the input is required to be 1-12). It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipment month (string)* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_ship\_ID("11",1)  Returns: Object { pass: true } | |
| **Function signature** | | **test\_ship\_quarter***(ship\_month, quarter, evalres,line)* |
| This function validates the shipment quarter and its matching with the entered shipment month. First it checks the presence of the quarter input. If an input (quarter) is entered, it checks for allowed characters (a numeric input is required), it also checks for the value range of the quarter (it is required to be between 1 and 4). Next, It validates the shipment month value, if no error is found, it checks if the ship month matches the correct entered quarter. The function returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected. | | |
| Argument : | * *ship\_month: shipment month (string)* * *quarter: shipment quarter (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_ship\_day***(input,line)* |
| This function validates the shipment day by checking its presence. If an input is entered, it checks for allowed characters (numeric input is required). It also checks for the value range. The range value check, checks for the range of values that are allowed to be entered (in this function the allowed range is 1-31). It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipment month (string)* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_ship\_day("28",1)  Returns: Object { pass: true } | |
| **Function signature** | | **test\_ship\_value***(input,line)* |
| This function validates the shipment value by checking its presence. If an input is entered, it checks for allowed characters (numeric input is required). It also checks for the value range. The range value check, checks for the range of values that are allowed to be entered (in this function the allowed range is greater than or equal 1). It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipment month (string)* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_ship\_value("15",1)  Returns: Object { valid: true, pass: true } | |
| **Function signature** | | **test\_ship\_weight***(weight, mode, naics, evalres,line)* |
| This function validates the shipment weight and its consistency with the entered shipping mode and NAICS code. First, it checks the presence of the input (weight). If an input (weight) is entered, it checks for the allowed characters (a numeric input is required), it also checks for the weight value range (minimum value is zero). If no error is found, it validates the entered shipping mode, if the entered shipping mode is valid, it checks if the weight is within the allowed range of the entered shipping mode. Next, it validates the entered NAICS, if the NAICS is valid, it checks if a list of NAICS (lkup30) and mode 4 of shipping are consistent with the entered weight. The function returns an object with three attributes, pass, valid and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected. | | |
| Argument : | * *weight: shipment weight (string)* * *mode: shipping mode (string)* * *naics: NAICS code (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_sctg***(sctg, value, weight, mode, temp, naics, state, evalres,line)* |
| This function validates the entered SCTG code and its consistency with the entered shipment value, shipping mode, temperature, NAICS code, and shipping state. First, it checks the presence of the input (SCTG). If an input (SCTG) is entered, it checks for the allowed characters (a numeric input is required), it also checks for the field length (it is required to have a length of 5), it also checks for the value range of the SCTG (it is allowed to be between 01001 and 43999). If no error is found, it validates the existence of the entered SCTG by running a binary lookup to check the existence of the SCTG in a lookup table (lkup1), if it exists, it validates both the shipment value and shipment weight. If no error is found, it checks the consistency of the weight and shipment value ratio with the SCTG code. Next, it validates the entered NAICS, if the NAICS is valid, it checks if the combination of SCTG and the entered NAICS is valid. Then it validates the entered shipping mode and weight, if they are valid, it checks for the consistency of them with the SCTG code. Finally, it validates the entered temperature, if it is valid, it checks its consistency with the SCTG code. The function returns an object with three attributes, pass, valid and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected. | | |
| Argument : | * *sctg: SCTG code (string)* * *value: shipment value (string)* * *weight: shipment weight (string)* * *mode: shipping mode (string)* * *temp: temperature controlled or not (string)* * *naics: NAICS code (string)* * *state: shipping state (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_sctg\_descr***(input,line)* |
| This function validates the description of the shipped commodity by checking its presence. If an input is entered, it checks for allowed characters (alphanumeric input is required). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipped commodity description (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_sctg\_descr("boxes",1)  Returns: Object { pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_temp\_control***(input,line)* |
| This function validates the shipment control temperature by checking its presence. If an input is entered, it checks the existence of the input in a lookup table (lkup25). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: y, n (string)* * *line: line number* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_temp\_control("y",1)  Returns: Object { valid: true, pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_unna***(unna, sctg, evalres,line)* |
| This function validates the UN or NA number and its consistency with the entered SCTG code. First, it checks the presence of the input (UN or NA number). If an input (UN or NA number) is entered, it checks for the allowed characters (only a numeric input is allowed), it also checks for the field length of the UN or NA number (it is required to be at length 4). Then, it checks for the value range of the UN or NA number, (ranging between 0004 and 9269). If no error is found, it validates the entered SCTG code, if the entered SCTG code is valid, it checks if the combination of UN or Na number is consistent with the SCTG code. The function returns an object with three attributes, pass, valid and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected. | | |
| Argument : | * *unna: UN or NA number (string)* * *sctg: SCTG code (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_destinationCity***(input,line)* |
| This function validates shipment destination city by checking its presence. If an input is entered, it checks for allowed characters (alphabetic input is required). It also checks for invalid characters by checking if the input has any not allowed input. It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipment destination city (string)* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_destinationState***(input,line)* |
| This function validates the shipment destination state by checking its presence. If an input is entered it checks for allowed characters (alphabetic input is required), it also checks for input length (only an abbreviation of two letters is allowed). Then, it checks for invalid characters. Finally, it checks for the existence of the input in a look-up table (lkup32). It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipment destination city (string)* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_destinationState("OH",1)  Returns: Object { valid: true, pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_destinationZip***(city, state, zip, evalres,line)* |
| This function validates the 5-digit mailing zip code of shipment destination by checking its presence. If a zip code is entered, it checks for allowed characters (it is required to be numeric). Then, it checks the input length (required to be 5-digit length). If no errors were found, it checks the existence of the entered zip code in a lookup table. If the zip code exists, it validates the entered city and state values, if no error is found it checks if the zip code matches the entered city and state. It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *city: destination city (string)* * *state: destination state’s two-letter abbreviation* *(string)* * *zip: destination city’s zip code (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_mode***(input,line)* |
| This function validates the entered transportation mode by checking its presence. If an input is entered it checks for allowed characters (numeric input is required), if no error is found, it checks for input length (1-3 field length is required). If no errors were found, it checks for the existence of the shipment mode in a look-up table (lkup3). It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: transportation mode (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, valid: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_mode("1",1)  Returns: Object { valid: true, pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_export***(input,line)* |
| This function validates if the shipment is exported or not by checking the input presence. If an input is entered, it checks for the existence of the input in a look-up table (lkup25). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: yes or no (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_export("y",1)  Returns: Object { pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_exportCity***(input,line)* |
| This function validates the shipment export city name by checking its presence. If an input is entered, it checks for allowed characters (alphabetic input is required). It returns an object with three attributes, pass, valid, and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: shipping city name (string)* * *Line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_exportCountry***(country, city, evalres, line)* |
| This function validates the entered export country name of a shipment by checking its presence. If a country name is entered, it checks for allowed characters (it is required to be alphabetic). If no error is found, it validates the city name, if no error is found, it validates the combination of the city and country names (for Canada and Mexico). If the export countries were not either Canada or Mexico, it only checks the existence of the entered export country in a look-up table. It returns an object with three attributes, pass, valid and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *country: export country name (string)* * *city: export city name (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_exportMode***(exp\_mode, country, evalres, line)* |
| This function validates the entered export mode of a shipment by checking its presence. If an export mode is entered, it checks for allowed characters (it is required to be numeric). If no error is found, it validates the country name, if no error is found, it checks the existence of the entered country name in a lookup table. If no error is found, it validates the combination of export mode with the export country (truck export mode is allowed only for Canada and Mexico). It returns an object with three attributes, pass, valid and errors. Pass and valid are binary attributes indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *exp\_mode: export mode number (string)* * *country: export country name (string)* * *evalres:* * *line: line number (integer)* | |
| Returns: | Object { valid: Boolean, pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_auto\_fill\_m***(input,line)* |
| This function runs an auto-fill test for a list of attributes which are: SHIPMENT\_VALUE, SHIMPENT\_WEIGHT, SCTG\_COMMODITY\_CODE, DOMESTIC\_TRANSPORT\_MODE, HAZMAT\_CODE, DOMESTIC\_ZIP\_CODE. It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: the input of the listed above attributes* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_auto\_fill\_m("1",1)  Returns: Object { pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_naics***(input,line)* |
| This function validates the NAICS code by checking its presence. If an input is entered it checks for allowed characters (numeric input is required), it also checks for input length (minimum length is 2 maximum length is 6). Next, it checks for the range of the input value (the minimum range is 11 and the maximum range is 928120). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: NAICS code (string)* * *line:* line number (integer) | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_naics("11",1)  Returns: Object { valid: true, pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_MOS***(input,line)* |
| This function validates the MOS code by checking its presence. If an input is entered it checks for allowed characters (numeric input is required). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: MOS code (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_MOS("1",1)  Returns: Object { valid: true, pass: true } | |

|  |  |  |
| --- | --- | --- |
| **Function signature** | | **test\_ATV***(input,line)* |
| This function validates the ATV code by checking its presence. If an input is entered it checks for allowed characters (only numeric input is allowed), and it checks for the allowed range of the input value (minimum range is 0 and maximum range is infinity). It returns an object with two attributes, pass, and errors. Pass is a binary attribute indicating whether the input has any errors. It is set to true if no error is found, otherwise, it is set to false. Errors is an array of objects reflecting the list of detected errors. | | |
| Argument : | * *Input: ATV code (string)* * *line: line number (integer)* | |
| Returns: | Object {pass: Boolean, errors: [ Object {flag: String, flagmsg: string, flagname: string, flagval: int, line: int, priority: int }]} | |
| Example: | Input type: test\_ATV("11",1)  Object { valid: true, pass: true } | |