

A stylized graphic on the left side of the slide. It features a globe with a network of lines connecting various points. Two satellite icons are positioned on the globe's surface, and two aircraft icons are shown in flight. Several triangles of varying sizes are scattered around the globe.

pyQuARC:

Open Source Library for Earth Observation Metadata Quality Assessment

December 17, 2021

Presenter: Jenny Wood

Co-authors: Jeanné le Roux, Slesa Adhikari, Ashish Acharya, Prasanna Koirala, Iksha Gurung, Camille Woods, Samuel Ayers, Bradley Baker, Emily Foshee, Andrew Weis, Stephen McNeal, Shelby Bagwell, Kaylin Bugbee, Rahul Ramachandran



**ARC Review
Process**

1

**pyQuARC
Development**

2

Agenda

3

Long Term Goals

ARC Review Process

- High-quality metadata is essential for data discoverability, accessibility, and usability. The ARC (Analysis and Review of CMR) project aims to improve these qualities by evaluating the completeness, correctness, and consistency of NASA's collection of over 8,000 Earth observation data products stored in the CMR (Common Metadata Repository).
- ARC has developed a metadata quality framework to assess these elements and identify opportunities for improvement. ARC's priority classification scheme (Fig. 2) is heavily implemented within this framework to identify metadata elements that need more attention.
- In order to streamline the review process, ARC is currently developing a host of scripts, known as pyQuARC to automate metadata quality assessment as much as possible.

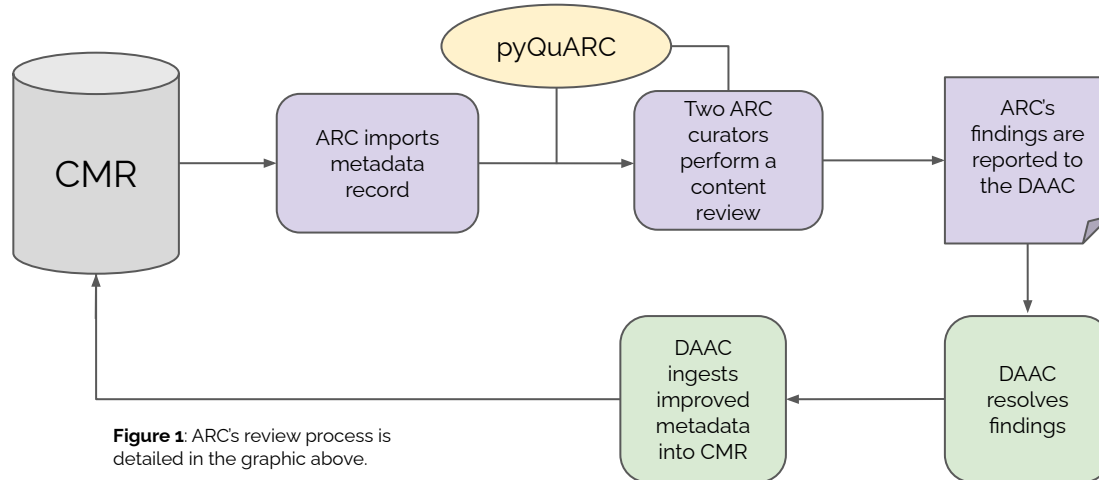


Figure 1: ARC's review process is detailed in the graphic above.

High [Error]	• Inaccurate, incomplete, or missing content • Broken URLs and invalid collection-granule relationships
Medium [Warning]	• Revisions of existing content • Addition of new information
Low [Info]	• Minor consistency issues

Figure 2: ARC's priority matrix is shown above.

pyQuARC Development

- Goal: implement ARC's metadata quality assessment framework in an automated tool
- The steps to develop pyQuARC are outlined in Figure 3.

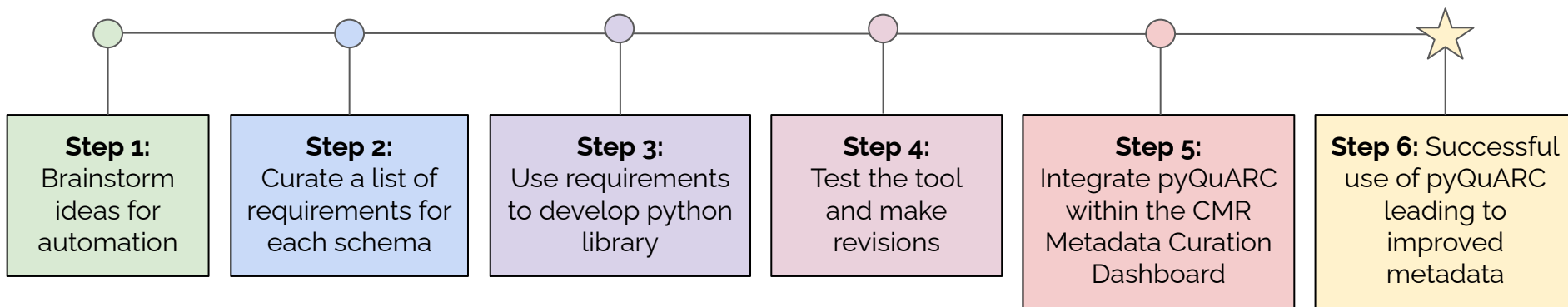


Figure 3: Steps of pyQuARC development.

pyQuARC Development

Step 1: Brainstorming

- By executing pyQuARC on metadata, it should discover high, medium, and low priority findings automatically
 - Note: what is considered high, medium, and low is documented quality criteria within ARC's framework

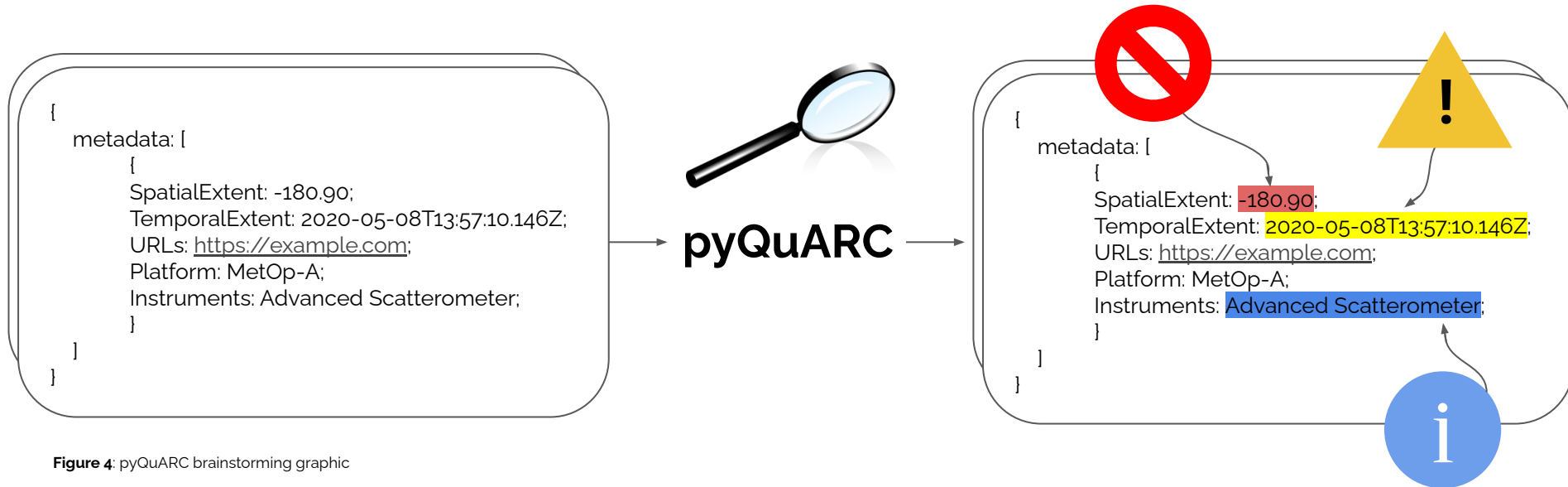


Figure 4: pyQuARC brainstorming graphic

pyQuARC Development

Step 2: Curate a list of requirements for each schema.

- Includes a list of each metadata field within the schema
- Describes pass/fail check(s) to perform on each element
- Assigns level of priority and associated output message if check fails

Note: pyQuARC is designed for CMR supported metadata schemas for Earth Observation data, but with enough customization, pyQuARC has the capability to support additional schema types.

Field	Check(s)	Priority	Message
Platform Short Name	GCMD check	Error	The provided platform short name does not comply with GCMD. Please submit a request to support@earthdata.nasa.gov to have this platform added to the GCMD Platforms KMS.
Horizontal Datum Name	Horizontal Datum Name Presence Check	Info	If appropriate for the data set, recommend providing the horizontal datum name.

Figure 5: Requirement examples

pyQuARC Development

Step 3: Use requirements to develop python library

- Design package to include checks and output priority messages
- Maintain relational dependencies between checks
 - Example: Collection Progress and Ending Date Time
- Ensure that checks are customizable, so future metadata standards can be implemented

```
► python pyQuARC/main.py -h
usage: main.py [-h] [--query QUERY | --concept_ids CONCEPT_IDS
[CONCEPT_IDS ...]] [--file FILE | --fake FAKE] [--format
[FORMAT]]

arguments:
-h, --help            show this help message and exit
--query QUERY          CMR query URL.
--concept_ids          CONCEPT_IDS
[CONCEPT_IDS ...]    List of concept IDs.
--file FILE            Path to the test file, either absolute or
                        relative to the root dir.
--fake FAKE           Use a fake content for testing.
--format [FORMAT]      The metadata format (currently supported:
                        'echo10' and 'dif10')
```

Figure 6: pyQuARC help menu

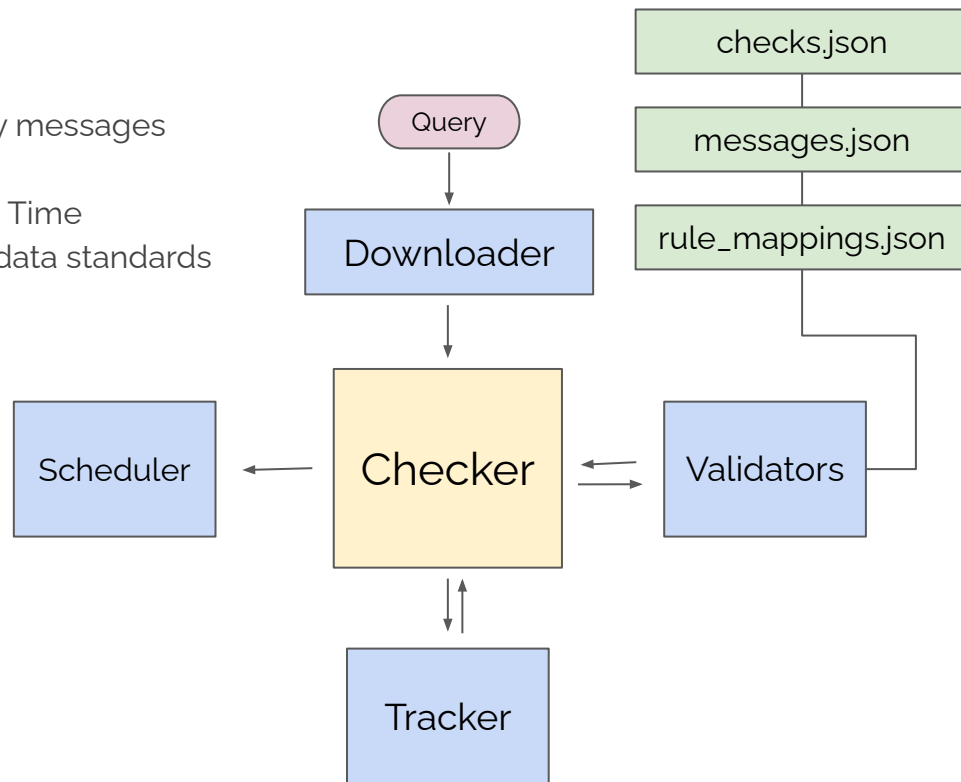


Figure 7: pyQuARC architecture diagram

pyQuARC Development

The checks, messages, and rule_mappings files work together to feed into the validators

Figures 8, 9, and 10 show a check example in each file.

checks.json

```
"campaign_short_name_gcmb_check": {  
  "data_type": "string",  
  "check_function":  
    "campaign_short_name_gcmb_check",  
  "available": true  
},
```

Figure 8: Campaign Short Name GCMD check in checks.json file

messages.json

```
"campaign_short_name_gcmb_check": {  
  "failure": "The provided campaign  
short name `{}` does not comply with  
the GCMD.",  
  "help": {  
    "message": "",  
    "url": ""  
  },  
  "remediation": "Please submit a  
request to support@earthdata.nasa.  
gov to have this campaign short name  
added to the GCMD Projects KMS."  
},
```

Figure 9: Campaign Short Name GCMD check in check_messages.json

rule_mappings.json

```
"campaign_short_name_gcmb_check": {  
  "rule_name": "Campaign Short Name  
GCMD Check",  
  "fields_to_apply": {  
    "echo10": [  
      {  
        "fields": [  
          "Collection/  
Campaigns/Campaign/  
ShortName"  
        ]  
      }  
    ]  
  },  
  "severity": "error"  
},
```

Figure 10: Campaign Short Name GCMD check in rule_mappings.json.

pyQuARC Development

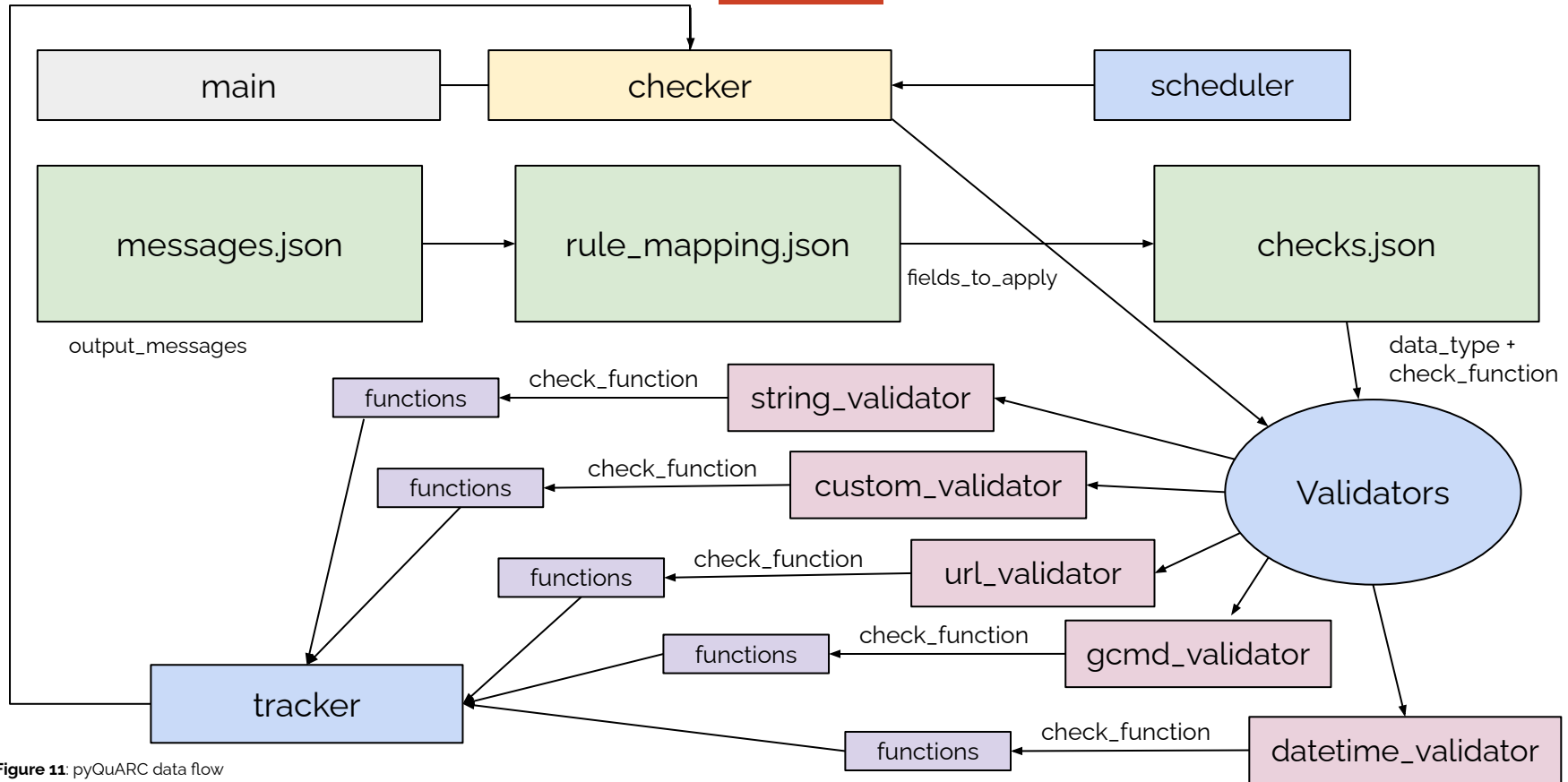


Figure 11: pyQuARC data flow

pyQuARC Development

Example pyQuARC output:

Figure 12: Example pyQuARC output

```
*****
** Metadata Validation Errors **
*****

METADATA: tests/fixtures/test_cmr_metadata.echo10

>> Collection/DeleteTime:
  Error: `2016-04-14` does not adhere to the ISO 1601 standard
  Make sure the datetime complies with ISO 1601 standard.

>> Collection/LastUpdate:
  Info: The UpdateTime `2016-04-14 00:00:00+00:00` comes before the provided InsertTime `2018-04-14 00:00:00+00:00`.
  Update the LastUpdate time so that it is identical to the InsertTime (in the event that the data has never been updated) or so that it comes chronologically after the InsertTime.

>> Collection/Temporal/SingleDateTime:
  Info: The Temporal requirements aren't met.
  Please provide at least one of SingleDateTime, RangeDateTime, PeriodicDateTime

>> Collection/Platforms/Platform/Instruments/Instrument/ShortName:
  Error: The provided instrument short name `TANSO-FTS` and long name `Thermal And Near Infrared Sensor For Carbon Observation` aren't consistent.
  Please supply the corresponding long name for the short name.

>> Collection/DOI/Explanation:
  Info: DOI Missing Reason Explanation is not provided.
  Please provide an explanation for why the DOI is not provided.

>> Collection/DOI/MissingReason:
  Error: `Some Reason` is not a valid value.
  The Missing Reason should read "Not Applicable".

>> Collection/CollectionState:
  Warning: CollectionState `COMPLETE` is not consistent with EndingDateTime/EndsAtPresentFlag.
  Please update the CollectionState based on the EndingDateTime/EndsAtPresentFlag values.

>> Collection/OnlineResources/OnlineResource/Type:
  Warning: Online Resource Type missing.
  Please provide a corresponding GCMD compliant Online Resource Type for each Online Resource URL.
```

pyQuARC development

Step 4: Test the tool and make revisions

- Modify test file and note inconsistencies between requirements and output
- Add/modify checks to resolve inconsistencies

Figure 13:
Original
echo10 rule
mappings
for Data
Update Time
Logic Check

```
"data_update_time_logic_check": {  
  "rule_name": "Data Update Time Logic Check",  
  "fields_to_apply": {  
    "echo10": [  
      {  
        "fields": [  
          "Collection/LastUpdate",  
          "Collection/InsertTime"  
        ],  
        "relation": "gte"  
      }  
    ],  
    "severity": "info",  
    "check_id": "date_compare"  
  },  
}
```



Figure 14: Revised rule mappings for Data
Update Time Logic Check to include dif10

```
"data_update_time_logic_check": {  
  "rule_name": "Data Update Time Logic Check",  
  "fields_to_apply": {  
    "echo10": [  
      {  
        "fields": [  
          "Collection/LastUpdate",  
          "Collection/InsertTime"  
        ],  
        "relation": "gte"  
      }  
    ],  
    "dif10": [  
      {  
        "fields": [  
          "DIF/Metadata_Dates/Data_Last_Revision",  
          "DIF/Metadata_Dates/Data_Creation"  
        ],  
        "relation": "gte"  
      }  
    ],  
    "severity": "info",  
    "check_id": "date_compare"  
  },  
}
```

pyQuARC Development

Step 5: Integrate pyQuARC within the CMR Metadata Curation Dashboard

- ARC reviewers use the CMR Metadata Curation Dashboard tool to make recommendations and report findings to DAACs
- Integrating pyQuARC within the Dashboard allows reviewers to more efficiently review metadata and note findings

Campaigns / Campaign / ShortName *	
REQUIRED BY SCHEMA AND/OR UMM	✓
SCRIPT RESULT	Errors: • Error: The provided campaign short name `IPY DATAPOOL` does not comply with the GCMD. • Error: The provided campaign short name `IPY GREENLAND L0` does not comply with the GCMD. Remediation: Please submit a request to support@earthdata.nasa.gov to have this campaign short name added to the GCMD Projects KMS.
CURRENT VALUE	• IPY DATAPOOL • IPY GREENLAND L0

Figure 15: Integrating pyQuARC within the CMR Metadata Curation Dashboard

Long Term Goals

- **Step 6:** Successful use of pyQuARC leading to improved metadata
 - pyQuARC is openly available and customizable allowing users outside of ARC to implement and restructure the tool to fit their needs. Examples include:
 - Configuring pyQuARC for additional schema types
 - Adding/removing checks
 - Adjusting output messages
 - With this level of customization, users within the broader NASA Earth science (EOSDIS) metadata community can utilize the tool for automatically assessing their own metadata records to ensure that Earth observation datasets are discoverable, accessible, and usable to the end user.

Thank you.



Contact information: jenny.wood@uah.edu

pyQuARC GitHub: <https://github.com/NASA-IMPACT/pyQuARC>

Improving Discovery and Use of NASA's Earth Observation Data Through Metadata Quality Assessments:
<https://datascience.codata.org/articles/10.5334/dsj-2021-017/>