Google Season Of Docs 2021 @ NumPy

NAME: Ayush Verma

EMAIL: verma16.ayush@gmail.com

GITHUB: verma16Ayush

ALIAS (For final case-study reports): AVR16

LOCATION: Kanpur, Uttar Pradesh, India. (UTC + 5 : 30)

WORKING HOURS: 6:00 pm - 11:00 pm (UTC + 5:30) (flexible)

RESUME: Click Here

BLOG: avr16.medium.com

Professional Background And previous work examples

- Currently a sophomore at the National Institute of Technology (NIT), Hamirpur, Himachal Pradesh, India.
- As mentioned in the resume, as a part of my work with IoT-web planet, I have created course videos, homework assignment and their solutions. An example of which could be found here on my blog (contents have been modified a little to fit for a blog post).
- Created an issue #18811 to NumPy referring to the problem when building documentation after having built NumPy into a virtual environment threw a `"numpy not found, cannot build documentation…"`
- Created a pull request #18812 enhancing the document and closing the aforementioned issue. The improved page could be found here.
- I have a well-received technical blog that generates healthy traffic considering it is still very young. The blog could be found here.
- As an executive member of GLUG(GNU/Linux users group)-NITH, a college community of computer science enthusiasts, I have created monthly programming contests ‘Heuristics’. The codeforces group can be found here.
- As an executive member of CSEC-NITH, I have given talks at programming meets focused on introducing freshmen to computer programming.

High-level Restructuring and end-user focus

1. Overview

The user documentation of any software project is the primary authority a user relies on to get fact-corrected and reliable educative/referential resources about the said project. Good documentation not only signifies a healthy and sustainable development environment, but it is also the first point of contact for new users/contributors and is pivotal in teaching tips, tricks and quick fixes for any known issues about the project and thus often a centrepiece in community building.

Good structure in documentation saves support cost and maximises efficiency and speed in the development process for new features and weeding out bugs and issues and enhances ease of access and searchability of relevant information

As of the current state of the NumPy, the documentation has the skeletal outline as laid out by the Diataxis framework but it still lacks critical structure. As of writing this statement, the docs are lined with duplicate content, a lot of misplaced content and missing/hard to find basic information, thereby being the major fault line that could use some improvement through dedicated efforts. This leads to users such as myself turn to unofficial resources to resolve queries that often lack critical information.

2. Analysis

As referenced in NumPy's GSoD project proposal, NEP 44 and a brief discussion with community members, I was introduced to the Diátaxis framework. I researched a lot about the guidelines to get a grasp of how it would consolidate the current NumPy document infrastructure and tried to form a conclusion.

Diátaxis Framework

Diátaxis is a lightweight document building framework that lays down clear goal-oriented, pragmatic guidelines to writing and organising a documentation project that helps users interact with the project depending on what level of familiarity they are with the project and with what aim they are trying to get information about the project.

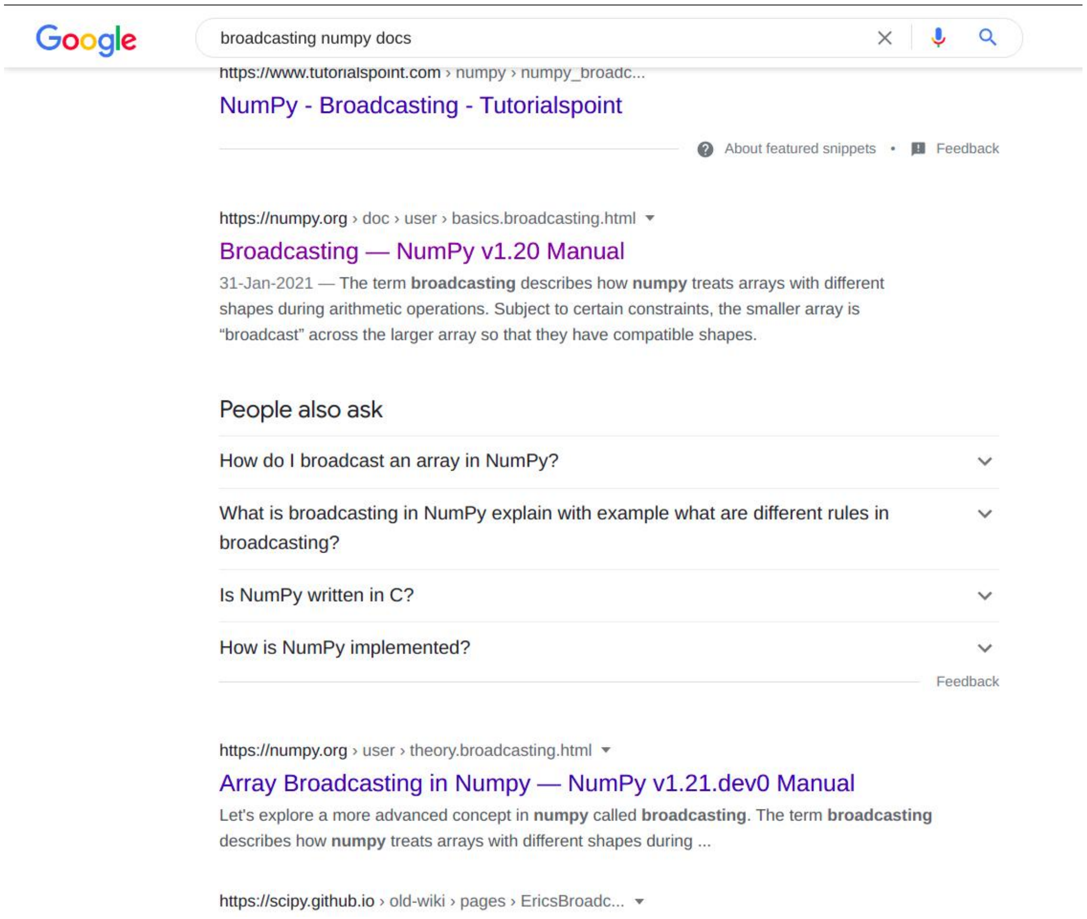
- It is a reliable and mature framework that has been the guiding hand behind the documentation projects of likes of Django and Gatsby. While Gatsby implements Diátaxis quite aggressively, Django does not shy away from departing from the framework to serve its users' best interests. NumPy's documentation structure could use somewhat an intermediate of these two, nearing towards that of Django.
- Diátaxis marks distinct boundaries between the 4 sections of any documentation project, as discussed with Melissa W. Mendonça over slack, I plan on following the Diátaxis sections as follows
    - Tutorials - for teaching basic techniques and best practices about NumPy.
    - How-Tos - for common questions related, but not confined, to doing frequent tasks with the project e.g. How to build NumPy and its documentation from source.
    - Reference - A clear and succinct description of what a piece of code or a function does **without** delving into how it does that. It answers to 'what... ?'.

- Explanations - A detailed yet concise explanation of various underlying concepts of the project that are fundamental to learning about NumPy

3. Issues

Below is a list of issues that I found while auditing the current documentation and its source. Although, please note that while I have tried to make the list as comprehensive as possible, for brevity's sake, it is still not exhaustive in the least of my findings. These are also more of a personal opinion than hard evidence of what is causing the "collapse of structure" and thus open to discussion with the mentors at the implementation stage.

- **Broadcasting**


broadcasting numpy docs

https://www.tutorialspoint.com › numpy › numpy_broadc...
NumPy - Broadcasting - Tutorialspoint

About featured snippets • Feedback

https://numpy.org › doc › user › basics.broadcasting.html
Broadcasting — NumPy v1.20 Manual
31-Jan-2021 — The term **broadcasting** describes how **numpy** treats arrays with different shapes during arithmetic operations. Subject to certain constraints, the smaller array is "broadcast" across the larger array so that they have compatible shapes.

People also ask
How do I broadcast an array in NumPy?
What is broadcasting in NumPy explain with example what are different rules in broadcasting?
Is NumPy written in C?
How is NumPy implemented?
Feedback

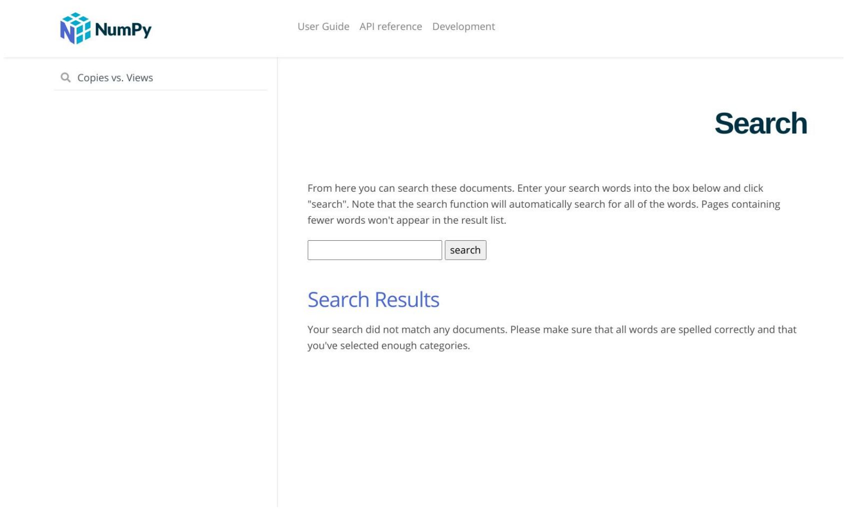
https://numpy.org › user › theory.broadcasting.html
Array Broadcasting in Numpy — NumPy v1.21.dev0 Manual
Let's explore a more advanced concept in **numpy** called **broadcasting**. The term **broadcasting** describes how **numpy** treats arrays with different shapes during ...

https://scipy.github.io › old-wiki › pages › EricsBroadc...

Google searching for broadcasting returns two different pages from the

NumPy's docs to here and here and some relevant content could also be found here. While one of the pages provides a crosslink to another, there is no need for one topic to live at two different locations

- **Copies vs. Views**
  I could not find any documentation on this topic on the web-hosted NumPy Docs. Google search and doc search results were as follows:

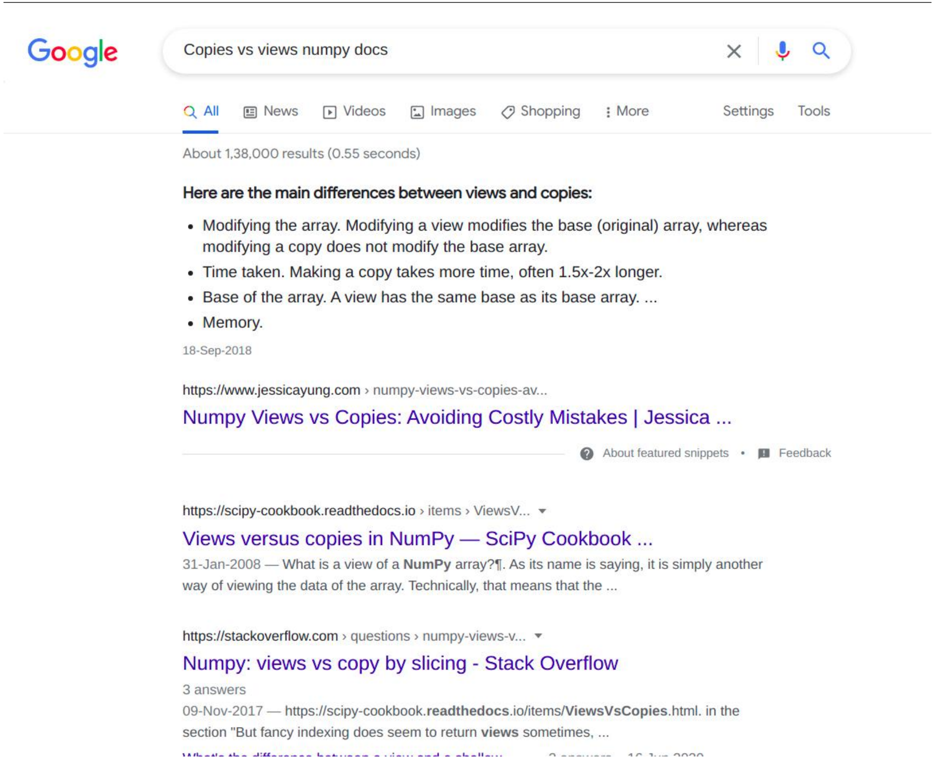


Search

From here you can search these documents. Enter your search words into the box below and click "search". Note that the search function will automatically search for all of the words. Pages containing fewer words won't appear in the result list.

Search Results

Your search did not match any documents. Please make sure that all words are spelled correctly and that you've selected enough categories.



About 1,38,000 results (0.55 seconds)

Here are the main differences between views and copies:

- Modifying the array. Modifying a view modifies the base (original) array, whereas modifying a copy does not modify the base array.
- Time taken. Making a copy takes more time, often 1.5x-2x longer.
- Base of the array. A view has the same base as its base array. ...
- Memory.

18-Sep-2018

https://www.jessicayung.com › numpy-views-vs-copies-av...
Numpy Views vs Copies: Avoiding Costly Mistakes | Jessica ...

https://scipy-cookbook.readthedocs.io › items › ViewsV...
Views versus copies in NumPy — SciPy Cookbook ...
31-Jan-2008 — What is a view of a **NumPy** array?¶. As its name is saying, it is simply another way of viewing the data of the array. Technically, that means that the ...

https://stackoverflow.com › questions › numpy-views-v...
Numpy: views vs copy by slicing - Stack Overflow
3 answers
09-Nov-2017 — https://scipy-cookbook.**readthedocs**.io/items/**ViewsVsCopies**.html. in the section "But fancy indexing does seem to return **views** sometimes, ...

Even a git grep returned only a reference to NEP44. The documentation for this topic could only be found in the pdf version of the NumPy 1.20 User Guide pg. 17. That either signifies either the documentation on this topic is missing or I could not find it, both more unfortunate than the other.

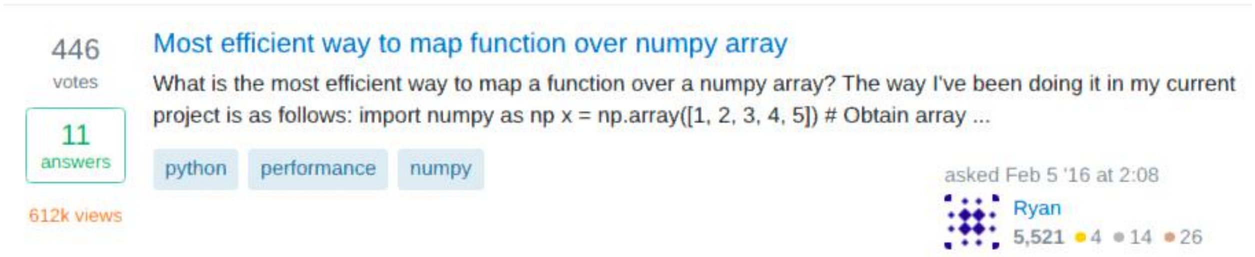
- **Tutorials**
  The tutorials section of the project needs major work and populating. Currently, there are two tutorials namely Linear Algebra on N-Dimensional Arrays and Masked Arrays. While these topics do a great job of teaching what they are advertising, they are somewhat non-beginner-friendly. The tutorials section should host at least one topic that could be accessible to users of all levels of expertise over the product. Anne Bonners' 'NumPy: the absolute basics for beginners' does an outstanding job at getting first-timers up to speed with NumPy, however, it does dive into data science concepts towards its end. I would like to add a couple of tutorials that could be followed by anyone with high school level math understanding. A few examples could include one or more from:
  - A tutorial on fancy indexing
  - Implementing McLaurin or Taylor series of e^x with NumPy
  - Implementing Conway's Game of life with only NumPy as referenced in NEP 44
  - Tutorial on writing Numpy tutorials
  - I/O with NumPy currently residing under NumPy Basics could also be migrated to NumPy tutorials (sections of which could also belong to a How-to guide, this is conflicted as of now for me)

- **How-Tos**
  This is another section of NumPy documentation that requires populating and expanding. Some topics that could be added to this section are:
  - Loading and storing data in various formats
  - Parallelization

  Apart from these some content from NumPy basics could also be migrated as a How-To guide like Byte-swapping and Subclassing ndarray etc.

Since I am nowhere near a conditioned expert on complex NumPy How-Tos, I turned to StackOverflow to look for some most frequently asked questions about NumPy that could be placed in a NumPy How-To Guide. A Noteworthy few include:

➢ How to efficiently map a function over an array



446 votes | 11 answers | 612k views

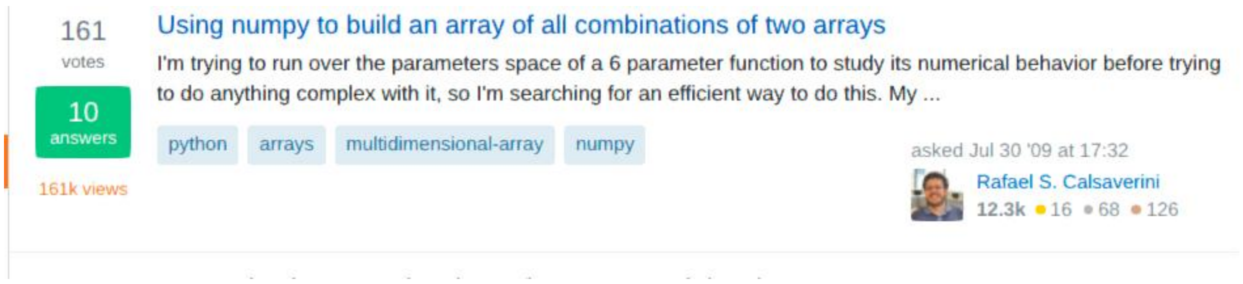
Most efficient way to map function over numpy array

What is the most efficient way to map a function over a numpy array? The way I've been doing it in my current project is as follows: import numpy as np x = np.array([1, 2, 3, 4, 5]) # Obtain array ...

python performance numpy

asked Feb 5 '16 at 2:08 Ryan 5,521 ●4 ●14 ●26

➢ Build an array of all combinations of two arrays



161 votes | 10 answers | 161k views

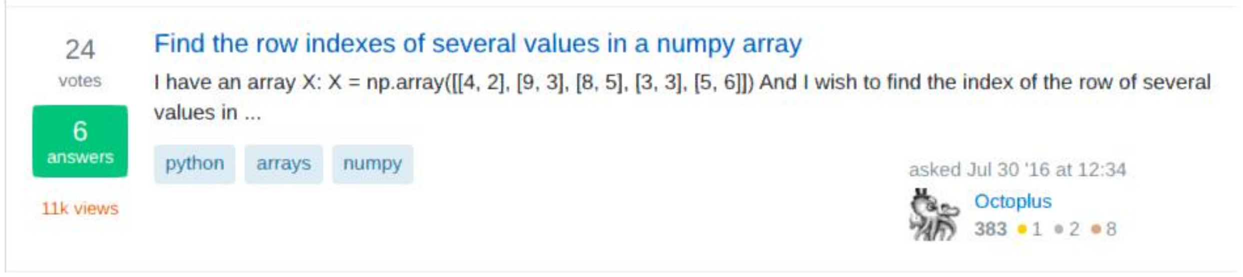
Using numpy to build an array of all combinations of two arrays

I'm trying to run over the parameters space of a 6 parameter function to study its numerical behavior before trying to do anything complex with it, so I'm searching for an efficient way to do this. My ...

python arrays multidimensional-array numpy

asked Jul 30 '09 at 17:32 Rafael S. Calsaverini 12.3k ●16 ●68 ●126

➢ Find row indices of several values



24 votes | 6 answers | 11k views

Find the row indexes of several values in a numpy array

I have an array X: X = np.array([[4, 2], [9, 3], [8, 5], [3, 3], [5, 6]]) And I wish to find the index of the row of several values in ...

python arrays numpy

asked Jul 30 '16 at 12:34 Octoplus 383 ●1 ●2 ●8

- **Explanations and References**
  The Reference section of NumPy docs is quite full and complete and in sync with the development of NumPy itself. However, It consists of numerous under-the-hood details that otherwise should belong to Explanations. A few examples could be:
  ➢ Copies vs. Views.

- ➢ Internal memory layout of ndarray from here in API reference to explanations.
- ➢ Introductory explanation to dtype from here in API reference to explanations.
- ➢ Introductory content from Scalars under API reference could also be migrated to explanations, extent and/or possibility of which I would like to discuss.

To conclude, the How-Tos and Tutorials section of the docs is intensive on the content creation side of the project while Explanations and API Reference is intensive on restructuring and reorganising.

4. Timeline

Project Length - 5 Months (standard Length)

1. **Phase 0 (Present - May 17)**
   - Continue exploring the documentation codebase.
   - Build a firm and more robust understanding of the current structure.
   - Research and educate myself more on Sphinx technicalities and Diataxis ideologies.

2. **Phase 1 - Community Bonding and Planning/Auditing (2 weeks)**
   - Get acquainted with the mentors. Set up evaluation deadlines, meeting hours, resolving timezone issues.
   - Audit current documentation state and formulate a detailed report on areas that need migrating from one section to another, content that is missing and discuss possibilities of tutorials and How-To guides that could be added.
   - Research and make a report of NumPy user survey as referenced in the GSoD proposal.
   - Formulate a broad layout of how the resulting document structure will look at the end of the project based on the above reports.

- Get reviews and make modifications to the proposed document structure as per mentor feedback and get it accepted before implementation.

3. **Phase 2 - Restructure and Reorganisation (8 weeks)**
    - Weekly/Biweekly mentor meetings to ensure the work efforts do not deviate from the original goal
    - During each one of these meets/slack conversations, go over the content that needs migrating to a different section of the documentation in the following week and report the work that was done in the previous week.
    - Auditing the work done after the restructuring and rebuilding any broken crosslinks.
        - Restructuring/reorganising (6 to 7 weeks)
        - Auditing/examining and proofreading (1 to 2 weeks)

4. **Phase 3 - Creating new Content for tutorials and How-to Guides and explanations(7 + 1 weeks)**
    - Go over the content that was discussed to be added with the mentors and incorporate changes in the plan, if any.
    - Add at least 1 and at max 2 Tutorials based on suggestions as discussed on pages 6 & 7 or as discussed with the mentors.
    - Copies vs. Views page to explanations.
    - Add at least 2 How-To guides based on suggestions as discussed on pages 6 & 7 or as discussed with the mentors.
    - A ‘How to write a NumPy Tutorial’ page for ease of future contribution from community members
    - Proofread for any errors and typos. (1 week)

5. **Phase 4- Spare weeks (2 weeks)**
    - 2 spare weeks in case of any emergencies, untoward happening or miscalculations in the timeline being proposed as of now.
    - Focus all efforts on auditing the work that was done and start creating a final project report.

6. **Phase 5 - Final evaluations**
    - Write and submit a final case study report for the fifth and final evaluation
    - Write and submit a report of my experience as a participant of GSoD'21 at NumPy

5. Project Outcomes

- Appropriately placed content and an easy to navigate document structure that conforms to Diataxis guidelines of a goal-oriented, pragmatic approach to structuring that users of different experience level can easily interact with.
- Pages with fresh content in Tutorials, How-To and Explanation sections of the documentation.

Why do I want to work on this project?

The greatest of human inventions and breakthroughs amount to nothing if it cannot be taught to others to reap its benefits, this selfless act of making people learn and grow through the stuff we make is what has been driving the Open-source revolution since its inception. NumPy is one of the forerunners of this revolution. I was 17 when I wrote my first lines of Python and was totally dumbfounded upon being told that it has no braces or semicolons. I have since grown into working on slightly bigger personal projects and gone from a "total-noob" to "a-noob-who-kinda-sorta-knows what-he-is-doing”. This journey has had only one constant. Open-source software. So it is not just a responsibility, it is also a privilege to give back to what I have been so generously given. And what better way to give back to the community than to make learning more accessible and open to others by writing documentation.

Why am I the right person for this project?

- I have past experience at teaching and a genuine interest in making people learn through my works.
- I am quite capable of coherently organising and articulating ideas to get my point across.
- I am also quite comfortable with version control systems, thus carrying out regular push, fetch, amend would not be a learning curve that would waste precious time.
- I understand the needs of NumPy users and areas that cause problems because I am one of them myself.

Commitment

I do not have any other major engagement during the proposed period and if accepted as a technical writer at NumPy, I would be committed, in utmost sincerity, to the proposed timeline to the best of my abilities. Apart from GSoD, regular college coursework are the only other time-consuming aspect. however, I can guarantee that I can commit 5-6 hours of daily time to achieve the goals and deadlines set forth in this Statement of Interest.

References

- Diataxis
- The collapse of structure - Diataxis
- NumPy Docs
- NumPy enhancement proposal 44
- Gatsby: Documentation
- Django: Documentation
- NumPy: The absolute basics for beginners - Anne Bonners
- NumPy GSoD page: High-level restructuring and end-user focus
- StackOverflow: most frequent questions about NumPy