

Markdown 이란?

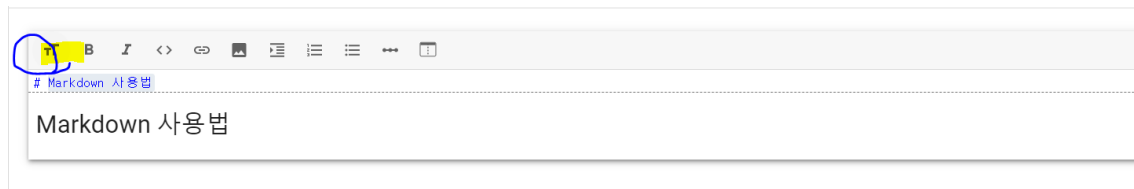
- 마크다운(markdown)은 일반 텍스트 문서의 양식을 편집하는 문법이다. README 파일이나 온라인 문서, 혹은 일반 텍스트 편집기로 문서 양식을 편집할 때 쓰인다. 마크다운을 이용해 작성된 문서는 쉽게 HTML, PDF, Word 등 다른 문서형태로 변환이 가능하다.
- 즉, Markdown은 텍스트를 작성하는 방법. 경량 마크업 언어이며, 간단히 HTML 형식으로 표현할 수 있는 특징이 있다.
- 그리고 직관적으로 읽기 쉽다는 장점이 있으며, 대표적으로 **GitHub**에서 **README.md** 파일을 작성할 때 자주 사용한다. (회사 내부의 Wiki에서 Markdown 형식을 사용하는 곳도 있습니다!)
- 즉, 강의할 때 제가 보여드리는 파일도 모두 markdown 언어를 이용해서 만든 것입니다. 장점은, 코드 블록이 예쁘게 잘 보이죠. indent도 그대로 유지되고요. 코드와 코드 결과를 보기에 매우 편리하고, 바로 html 형식 즉 인터넷 사이트로 전환이 되어서 -> 거의 대부분의 프로그래밍 보고서는 markdown으로 작성합니다.
- Colab에서는 [텍스트] 부분에 마크다운 언어의 문법을 이용할 수 있도록, 편리한 기능을 제공하고 있어요. 텍스트 셀 부분에 마크다운 문법을 그대로 이용하면 됩니다. 물론 버튼으로 간단하게 할 수 있도록 기능을 제공하고 있어요.
- 더 자세한 것을 알고 싶다면 = markdown 문법에 대해서 잘 알아두는 것은 여러모로 좋습니다.

<https://heropy.blog/2017/09/30/markdown/>

또는 코랩에서 제공하는 마크다운에 대한 설명인 다음 사이트를 참고하세요.

https://colab.research.google.com/notebooks/markdown_guide.ipynb

1. 제목 만들기



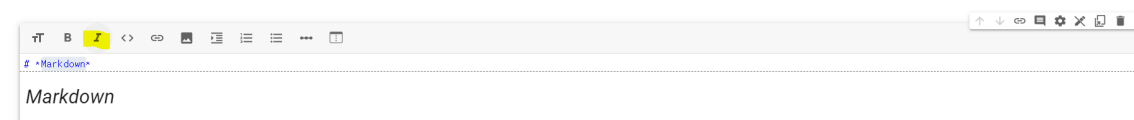
'#' 표시가 앞에 붙으면서 제목이 생성된다. # 의 개수가 많아질수록 글씨가 작은 제목이 형성된다.

2. 글씨를 진하게



원래 Markdown 문법에서 '** **' 를 붙이면 bold체가 된다. 그러나 간편하게 클릭으로 기능을 이용하도록 colab이 제공해준 것이다.

3. 글씨 기울이기 (기울임꼴)



4. 코드 블럭을 만들어준다. [코드로 형식 지정]

```
# Markdown
<pre>a = 2
b = 4
print(a+b)
</pre>
```

Markdown

```
a = 2
b = 4
print(a+b)
```

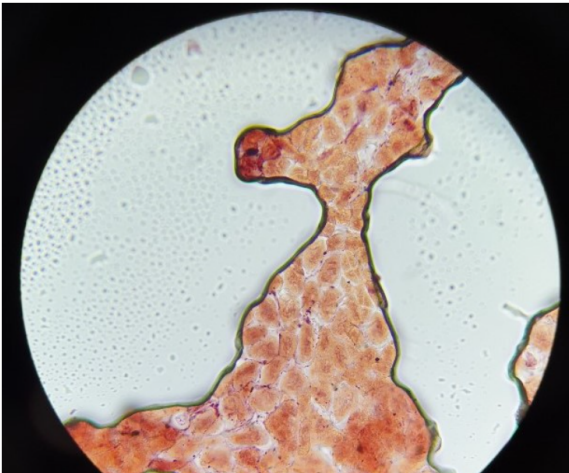
5. 홈페이지 링크, 보통 코드의 출처나 자료의 출처를 표기할 때 이용한다. [링크 삽입]

```
# Markdown[링크 텍스트](https://)
Markdown<a href="https://">링크 텍스트</a>
```

6. 이미지 첨부

```
# Markdown[링크 텍스트](https://)
Markdown<a href="https://">링크 텍스트</a>
```

Markdown



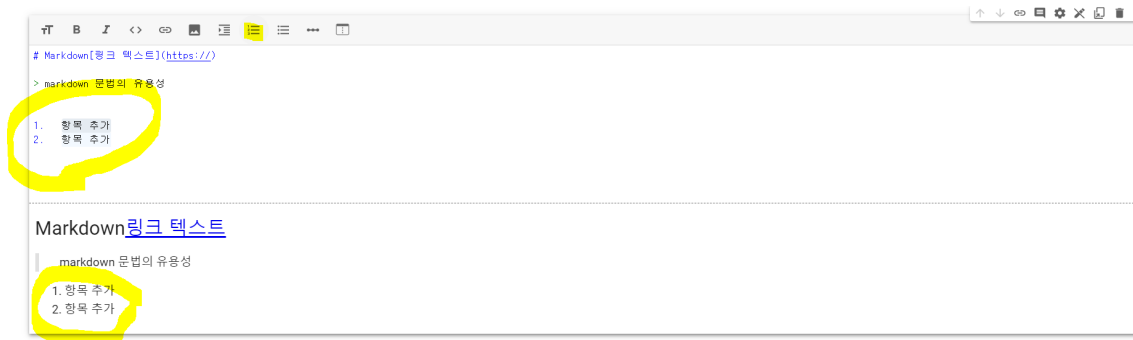
7. 한 칸 들여쓰기 (훨씬 보기 좋다.)

```
# Markdown[링크 텍스트](https://)
> markdown 문법의 유용성
|
```

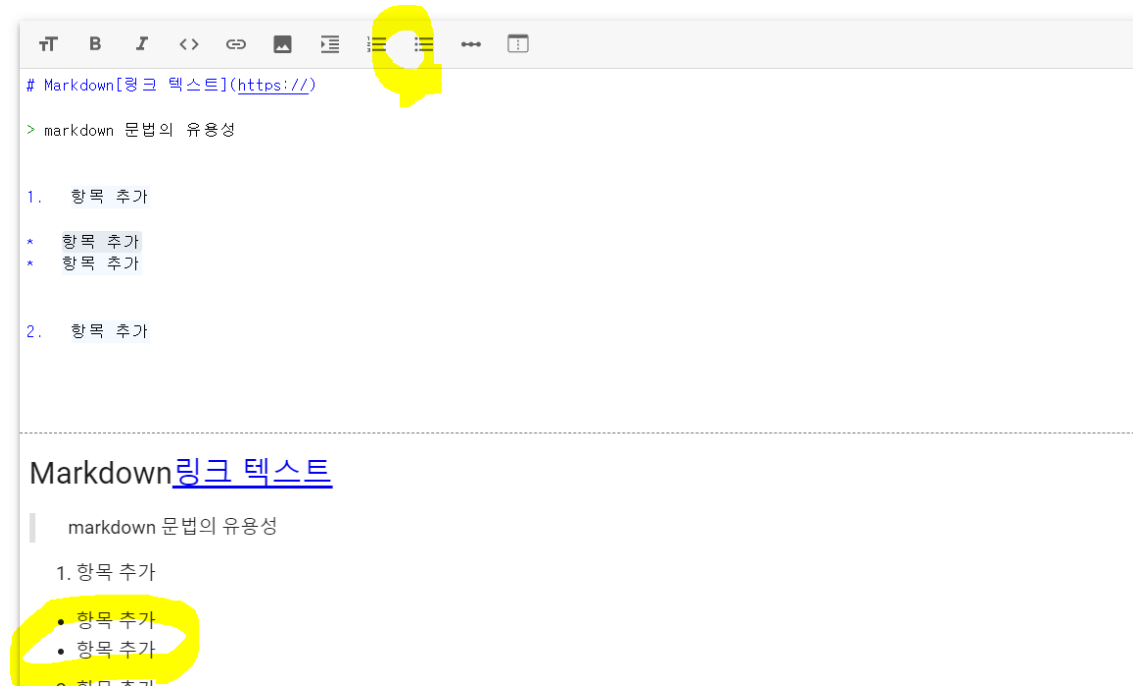
Markdown링크 텍스트

```
markdown 문법의 유용성
```

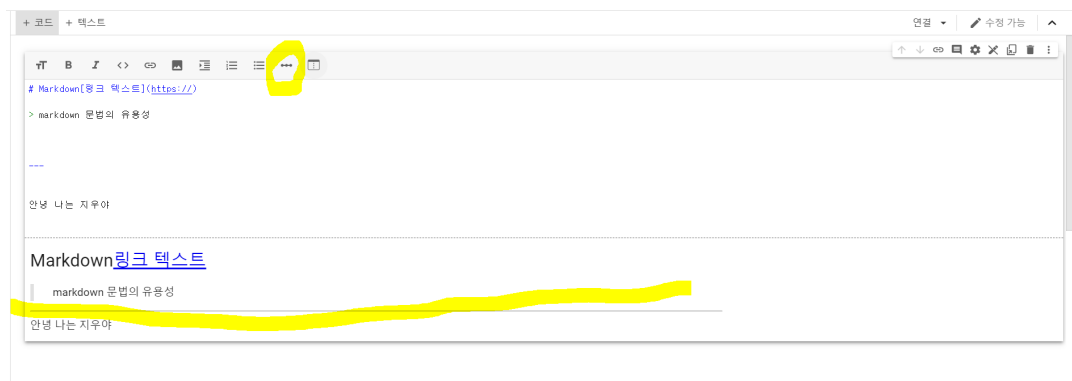
8. ordered list



9. unordered list



10. 수평선 긋기 - 코드를 분리할 때 많이 사용!



-> 보기에 편안하다.

- 중요한 점은 colab 파일의 가독성을 높이기 위해 셀과 셀 단위에 대해 제목을 붙여주고, 설명해주는 부분이 있어야 좋다는 것이다. 다음은 필자의 숙제 파일 일부이다.

파일 수정 보기 삽입 렌타일 도구 도움말 오후 12:43에 마지막으로 저장됨

≡ 목차 X + 코드 + 텍스트 연결 ▾ 수정 가능 ↗

▼ 선택지 목록

성능에 영향을 미칠 수 있는 parameter들과 몇 가지 예시를 적어보았습니다. 이들을 다양하게 테스트해보며 가장 높은 성능을 얻어내는 선택지를 도입하는 방식을 취했습니다.

Konlpy tagger의 종류

`tokenizer()` (1-1)에 사용하는 tagger의 종류에 따라 효율이 바뀔 수 있다. Pretrained word2vec embedding에서 사용된 tagger를 이용하는 것이 의 빈도를 줄여 더 의미있는 값을 가져오리라 기대할 수 있다.

- Hannanum
- Kkma
- Mecab
- Komoran
- Okt (구 Twitter)

또한, 어떤 형태소 집합을 분석에 사용할 것인지를 비롯한 `tokenizer()`의 다른 요소들도 결과에 유의미한 영향을 미칠 수 있다.

nn.Module의 종류와 Parameter

Neural network module을 어떤 것을 사용할 지에 따라 training되는 속도나 최종 accuracy가 달라질 수 있다.

- FNN (w/o TF-IDF scoring)
 - (TF-IDF scoring 방법)
 - Nonlinear FC layer의 개수
 - Hidden layer의 dimension

이러한 방식으로 목차와 설명과 코드를 분리해서 깔끔하게 작성해주자.