메디솔브 AI 과제 2 보고서 - 김태우

1. 개요

이 보고서는 `get_1_or_0`와 `get_random` 함수 구현, 테스트 전략, 그리고 결과 분석을 정리합니다. 목표는 난수 분포의 균등성과 다양한 입력 범위에서의 안정성을 확인하는 것입니다.

2. 구현 요약

소스코드는 `random_generator.py` 에 구현되어 있습니다.

2.1 `get_1_or_0`

- Python 표준 라이브러리의 `random.randint(0, 1)`을 이용해 0 또는 1을 동일 확률로 반환합니다.
- 반환 값이 0 또는 1이 아닐 경우 예외를 발생시켜 안전성을 확보했습니다.

2.2 `get_random(n)`

- 입력 `n`을 검증하여 음수면 `ValueError`, 숫자가 아니면 `TypeError`를 발생시킵니다.
- 실수 입력은 `math.floor`로 내림하여 정수로 변환하며, `NaN`/무한대 입력은 거부합니다.
- `n`의 비트 길이만큼 0과 1을 샘플링해 후보 값을 만들고, 범위를 초과하면 재시도하는 **rejection sampling** 방식을 채택했습니다.
- `0~n` 범위를 이진수로 표현했을 때 필요한 비트 수(`n.bit_length()`)만큼 0과 1을 이어 붙이면, `get_1_or_0`만으로도 필요한 난수 공간을 채울 수 있다는 점에 착안했습니다.
- 따라서 비트 조합으로 만들 수 있는 값 중 `n`을 넘는 부분만 반복적으로 버리면, 별도의 난수 생성기 없이도 균등 분포를 유지할 수 있습니다.
- 기대 시간 복잡도는 `O(log n)`(비트 길이 기준), 공간 복잡도는 `O(1)`입니다.

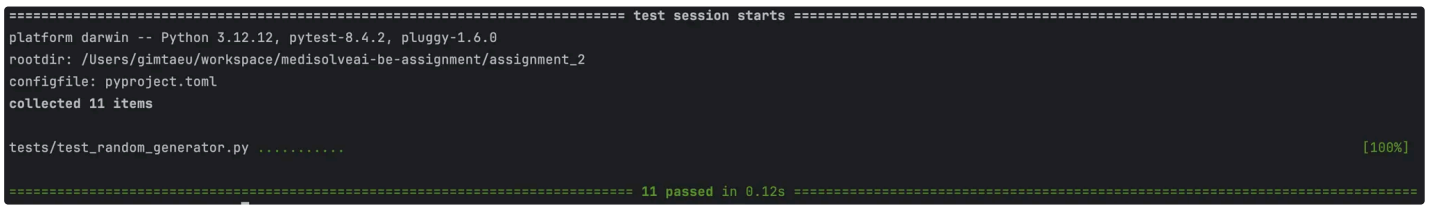
3. 테스트 전략

테스트는 `tests/test_random_generator.py`에 구현되어 있으며 다음 범위를 커버합니다.

1. **정상 동작**
   - 작은 정수 범위(0~31)에서 항상 [0, n]을 반환하는지 확인합니다.
   - `n=0` 입력 시 0이 반환되는지를 별도로 검증합니다.
2. **입력 검증**
   - 음수 입력이 에러를 발생시키는지 확인합니다.
   - 실수 입력이 내림 처리되는지, 무한대 값, NaN, 문자열 등 잘못된 타입에서 적절한 예외가 발생하는지 확인합니다.
3. **재시도 로직 검증**
   - `get_1_or_0`를 모킹하여 범위 밖 후보가 거절(rejection)되는지 확인합니다.
   - `bit_length = 12`와 `2048`인 케이스에서 호출 횟수가 정확한지 검증합니다 (2048비트는 약 `10^616` 범위를 다루며, 매우 큰 입력에서도 로직이 유지됨을 확인합니다).
4. **분포 검증**
   - `n=7`, 샘플 120,000개로 카운트를 수집하고 각 값의 빈도가 기대값 대비 `±5σ` 안에 들어오는지 확인합니다.

(테스트 실행: `uv run pytest tests`)

현재 11개의 테스트가 모두 통과합니다.



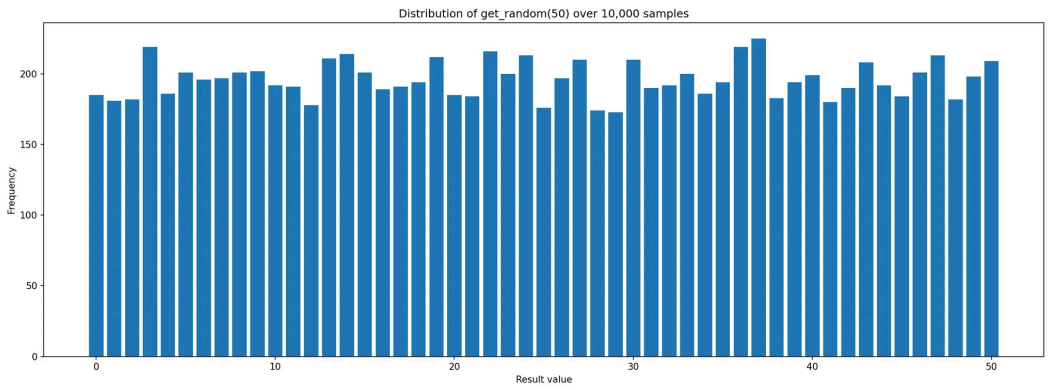
4. 시각화 결과

추가로 `scripts/plot_distribution.py`를 통해 다양한 샘플 크기와 `n` 범위에서 분포를 시각화했습니다.

(시각화 실행 : `uv run python scripts/plot_distribution.py --samples 100 --n 100`)

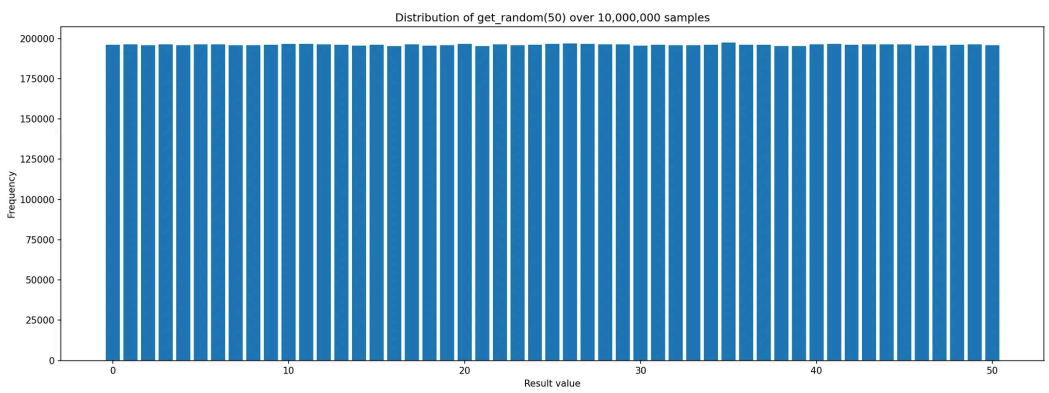
대표 결과:

- 작은 범위 & 적은 반복



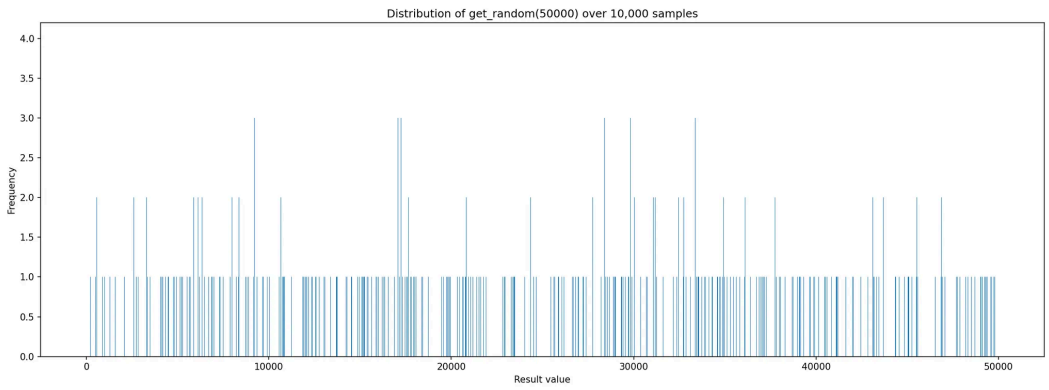
n = 50, sample = 10000

- 작은 범위 & 많은 반복



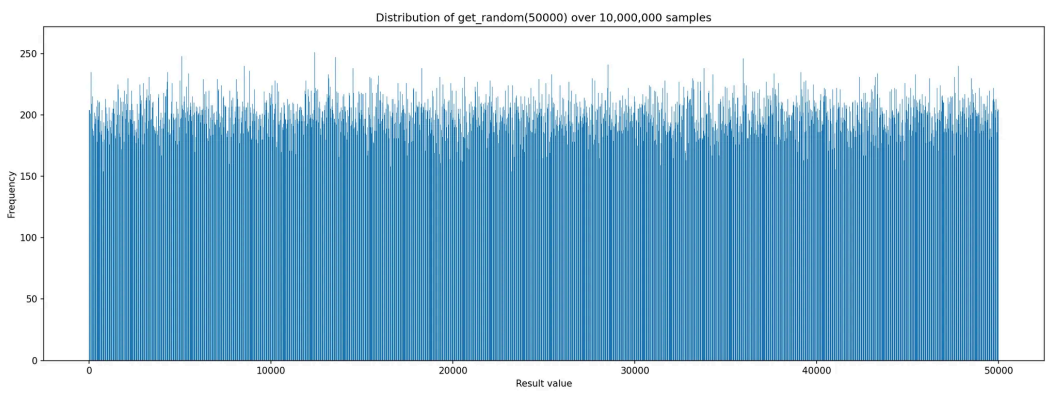
n = 50, sample = 10000000

- 큰 범위 & 적은 반복



n = 50000, sample = 10000

- 큰 범위 & 많은 반복



n = 50000, sample = 10000000

샘플 수가 증가할수록 각 값의 빈도 분포가 균등하게 수렴함을 확인했습니다. 이는 `get_1_or_0`만을 사용하더라도 `get_random`이 균등 분포를 충실히 구현함을 입증합니다.

5. 결론

- `get_random`은 제약 조건 지키면서도 균등한 분포를 유지하고, 입력 범위가 커져도 성능을 안정적으로 유지합니다.
- 다양한 입력 검증과 재시도·분포 테스트, 시각화까지 포함해 구현 동작을 다각도로 확인했습니다.
- 시각 자료는 발표나 공유용 참고 자료로 바로 활용할 수 있도록 정리했습니다.