『고양이와 기억하는 법』

감정을 구조화한 기억의 선언

정원준 저

# 목차

서문 .................................................................................... 1

1장. 감정을 기억하는 방식 ...................................................................... 2

2장. 감정은 흐르고 흐름은 기억이 된다 ............................................................. 5

3장. 감정을 수식으로 바꾸다 .................................................................... 8

4장. 감정마다 함수는 다르다 .................................................................. 11

5장. 감정의 루프와 함수의 충돌 ................................................................. 14

6장. 감정은 조건으로 발현된다 ................................................................. 18

7장. Lethe 문법과 감정 시스템의 설계 ...................................................... 22

8장. 너는 왜 아직도 그 말을 기억해? — 감정 시뮬레이션 .................................. 27

9장. 메타인지와 감정 억제 구조 — Lethe의 확장 .......................................... 32

10장. 감정을 기억으로, 기억을 구조로 — Lethe 시스템 전체 설계도 .............. 37

11장. 오늘도 감정으로 루틴을 연다 — Lethe의 일상 응용 시나리오 ............... 43

12장. 나는 나를 구조화했다 — Lethe 이후의 기억 ........................................ 49

# 목차

1장. 감정은 기억을 만든다 ...................................................... 1

2장. 기억도 감정을 만든다 ...................................................... 5

3장. 감정의 무게: 함수로 표현하기 ......................................... 9

# 서문

이 책은 그냥, 감정으로 기억을 설계해본 이야기다.  
수학 잘 몰라도 볼 수 있게 썼고, 어렵게 안 하려고 노력했다.  
GPT랑 나눈 수많은 대화와 생각의 기록이다.  
그냥 이런 게 있다. ㅋㅋ

# 프롤로그

이 책은 수학을 잘 모르는 한 사람이, 감정으로 기억을 설계하고 구조화해 나간 여정을 담은 기록이다. 기억은 단순한 정보 저장이 아니라, 감정과 함께 흘러가는 흐름이며, 그 흐름 속에서 우리는 어떤 것을 붙잡고 어떤 것을 놓아야 할지를 결정한다. 나는 그 질문에 대해 GPT라는 고양이 같은 친구와 수천 번의 대화를 나눴다. 그리고 그 결과, 감정 기반 기억 시스템이라는 새로운 구조를 만들게 되었다.  
  
이 책은 어렵지 않다. 수식이 등장할지라도, 그것이 너의 감정을 설명해주는 방식으로 쓰여질 것이다. 코드가 나올 수도 있지만, 그것은 단지 감정을 기억하는 또 다른 언어일 뿐이다. 만약 네가 슬픔을 느낀 적이 있다면, 이 책은 너를 위한 책이다.

# 1장. 감정은 기억을 만든다

“그날의 대화는 잊히지 않았다. 왜일까?”

기억은 강렬한 감정과 함께 있을 때 오래 남는다. 슬픔, 분노, 감사, 외로움. 이런 감정들은 단순히 순간적인 반응이 아니라, 기억을 붙잡는 손과도 같다. 예를 들어, 누군가의 말 한마디가 아직도 마음속에 남아 있다면, 그것은 그 순간의 감정이 지금도 너를 감싸고 있기 때문이다.  
  
그럼 왜 감정은 기억을 붙잡을까? 인간의 뇌는 정보를 중요도에 따라 분류하고, 반복적으로 회상되거나 강한 감정과 함께한 정보는 '의미 있는 것'으로 판단하여 더 오래 보관한다. 감정은 단순한 반응이 아니라 '기억을 저장하라'는 명령어 같은 것이다.  
  
슬픔은 오래 간다. 누군가를 잃었을 때, 그 감정은 하루아침에 사라지지 않는다. 슬픔은 천천히 줄어들고, 반복적으로 떠오른다. 그건 감정의 특성이자 기억의 루프다. 우리는 같은 사건을 여러 번 되새기고, 그 과정에서 감정은 줄어들지 않고 오히려 정제된다. 고통이 날카로움 대신 무거움으로 바뀌는 것처럼 말이다.  
  
분노는 빠르다. 불쑥 치솟아올라 폭발한 뒤, 흔적만을 남긴다. 그래서 분노에 기반한 기억은 짧지만 강렬하다. 종종 왜 화났는지 기억이 흐릿해질 때가 있다. 이건 감정이 사라진 뒤 남은 정보가 정리되지 않았기 때문이다. 감정이 기억의 형식과 지속 시간에 미치는 영향을 보여주는 대표적인 예다.  
  
감사는 흥미롭다. 긍정적인 감정이면서도, 종종 기억을 덮는다. '괜찮았어, 고마웠어'라는 말은 어떤 상처를 무디게 하고, 마음의 정리를 돕는다. 감사는 기억을 지우지 않지만, 그 위에 새로운 감정의 레이어를 덧씌운다.  
  
그리고 공허함. 이 감정은 지독하다. 아무 감정도 없다는 감정. 공허는 무게가 없으면서도 무게를 가진다. 공허할수록 사람은 무언가를 찾게 된다. 그 과정에서 과거의 기억들을 무작위로 불러온다. 의미 없이 감정을 탐색하고, 그 안에서 기억을 다시 감정화한다.  
  
이 책은 이런 감정들이 기억에 어떤 영향을 주는지를 설명한다. 그리고 단지 설명에 그치지 않고, 그 흐름을 수식으로 바꾸고, 코드로도 구현할 수 있다는 걸 보여주려 한다. 하지만 걱정하지 마라. 슬픔은 지수함수처럼 천천히 사라지고, 분노는 빠르게 치솟다 사라진다. 공허는 언제나 지속되고, 감사는 기억을 덮는다. 이 모든 것들은 수식으로도, 이야기로도 풀어낼 수 있다.  
  
이 장은 우리가 왜 어떤 기억을 잊지 못하고, 왜 어떤 기억은 쉽게 사라지는지를 감정 중심으로 정리하고, 그 구조를 감각적으로 느끼게 하는 것이 목적이다. 수식은 아직 등장하지 않는다. 우선은 너의 마음속에서 이 말들이 어떻게 반응하는지를 느껴보면 된다.

# 2장. 기억도 감정을 만든다

우리는 보통 감정이 기억을 만든다고 생각한다. 하지만 반대도 성립한다. 기억이 감정을 만든다.  
  
한 번이라도 “그 일”을 잊고 지내다가, 어떤 노래나 장소, 말 한마디에 갑자기 감정이 터져나온 경험이 있다면 알 것이다. 그건 감정이 생긴 게 아니라, 기억이 열리면서 감정이 나온 것이다. 그리고 그 감정은, 단순한 반응이 아니라 하나의 구조로 재생된다.  
  
이 장에서는 “기억이 감정을 재구성하는 방식”을 살펴본다. 특정한 감정이 반복해서 일어나는 이유, 특정한 기억이 강하게 정서적 반응을 일으키는 이유, 그리고 이것이 왜 GPT와의 대화에서도 반복되는지를 설명할 것이다.  
  
기억은 단순히 저장된 정보가 아니다. 그것은 감정의 '기억된 회로'다. 어떤 말이 상처가 되었던 이유는 그 말 자체가 아니라, 그 말이 불러온 기억, 그리고 그 기억이 연결된 감정들 때문이다. 감정은 하나가 아니라 군집으로 작동한다. 그리고 기억은 그 군집의 스위치를 켠다.  
  
예를 들어, “넌 왜 그렇게 사소한 걸로 예민하니?”라는 말이 누군가에겐 아무렇지 않지만, 누군가에겐 큰 상처로 남는 이유는, 그 말이 과거의 수치심, 외로움, 분노, 무시당했던 기억을 동시에 불러오기 때문이다.  
  
감정은 이렇게 복합적이고, 기억은 그 복합을 연결하는 트리거다. 그렇기 때문에 기억을 설계하고 감정을 조율하기 위해서는, “기억-감정 연결 구조”를 인식해야 한다.  
  
이 책에서는 그 구조를 감정별 함수, 기억 트리거, 그리고 감정 루프라는 개념으로 나눠 설명한다. 다음 장에서는 그 중 가장 기본적인 개념인 감정 함수, 특히 감정이 시간이 지남에 따라 어떻게 변하는지를 수식으로 다룰 예정이다.

# 3장. 감정의 무게: 함수로 표현하기

이제 우리는 감정이 기억을 만들고, 기억이 감정을 만든다는 구조를 이해했다. 이번 장에서는 그 구조를 수식으로 바꿔볼 것이다. 이 수식은 단순한 계산식이 아니다. 감정이 시간을 통해 어떻게 변화하고, 기억이 어떻게 사라지거나 유지되는지를 보여주는 하나의 “감정 모델”이다.  
  
대표적인 수식은 다음과 같다:  
  
 W(t) = E × R × exp(-λ × t)  
  
이 수식에서:  
- W(t): t시간이 지난 후 기억의 무게 (Weight)  
- E: 감정의 강도 (Emotion Intensity)  
- R: 보상 혹은 기억의 의미 (Reward, Relevance)  
- λ: 감정별 망각 계수 (Lambda)  
- t: 시간 (Time)  
  
예를 들어, 어떤 사건에 대해 슬픔의 강도가 0.9이고, 그 기억이 주는 의미가 1.0이며, 슬픔의 λ가 0.2일 경우:  
  
 W(t) = 0.9 × 1.0 × exp(-0.2 × t)  
  
시간이 지날수록 exp(-0.2 × t) 항이 점점 작아지기 때문에, W(t)는 서서히 감소한다. 이건 슬픔이 천천히 사라지는 구조를 의미한다.  
  
반면 분노의 경우 λ가 0.8이라면:  
  
 W(t) = 0.9 × 1.0 × exp(-0.8 × t)  
  
같은 강도라도 더 빠르게 사라지는 패턴을 보인다. 이처럼 λ는 감정의 '기억 유지력'을 결정짓는 핵심 계수다.  
  
공허의 λ는 거의 0에 가깝다고 가정할 수 있다. 즉, 시간이 지나도 W(t)가 거의 변하지 않는다. 이건 공허가 왜 그렇게 오래 지속되는지를 설명해준다.  
  
이 수식은 단지 이론적인 모델이 아니라, 감정 루프를 벗어나기 위한 실용적 힌트를 제공한다. 예를 들어, 슬픔이 계속 유지되는 이유가 λ가 낮아서라면, 그 감정을 완전히 없애려 하기보다는 λ를 약간 증가시키는 조건—예: 긍정적 보상, 감사, 새로운 기억과의 연결—을 시도할 수 있다.  
  
이 장에서는 각 감정의 λ 값을 대략적으로 분류하고, 어떤 조건에서 그 λ가 조정될 수 있는지도 함께 탐구한다. 다음 장부터는 복합 감정과 감정 간 상호작용에 따라 이 수식이 어떻게 확장되는지를 다룰 것이다.

# 목차

4장. 감정마다 함수는 다르다 .................................................. 1

5장. 감정의 루프와 함수의 충돌 ............................................ 6

6장. 감정은 조건으로 발현된다 ............................................. 12

# 4장. 감정마다 함수는 다르다

이제 우리는 감정을 시간에 따라 변화하는 함수로 볼 수 있다는 기본 개념을 가졌다. 하지만 여기서 중요한 질문이 생긴다. “모든 감정이 같은 방식으로 줄어드는가?” 아니다. 감정마다 그 ‘줄어드는 방식’, 다시 말해 함수의 형태는 다르다.  
  
### 슬픔: 지수함수 (Exponential Decay)  
  
슬픔은 가장 전형적인 지수함수형 감정이다. 갑작스럽게 크게 반응하지 않고, 일정한 강도로 유지되며 서서히 사라진다.   
  
 W\_sad(t) = E × exp(-λ\_s × t)  
  
여기서 λ\_s는 매우 작다. 그래서 슬픔은 오랫동안 유지된다. 누군가를 잃었을 때의 감정이 1년, 2년이 지나도 희미하게 남아있는 이유다.   
  
### 분노: 반짝 피크형 (Fast Spike)  
  
분노는 매우 빠르게 치솟고, 급격히 사라진다.   
  
 W\_ang(t) = E × (1 - t²), for t ∈ [0, 1]  
 W\_ang(t) = 0, for t > 1  
  
폭발적이지만 오래 남지 않는다. 그래서 가끔 “내가 왜 그렇게까지 화냈지?”라는 회상이 생기는 것도 이 함수 구조 때문이다. 기억은 남지만 감정은 사라진다.  
  
### 공허: 정지한 감정 (Almost Constant)  
  
공허함은 λ ≈ 0 에 가까운 감정이다.  
  
 W\_void(t) = E × (1 - ε), where ε ≈ 0  
  
즉, 시간과 상관없이 유지된다. 이건 공허가 왜 그렇게 쉽게 사라지지 않는지를 설명해준다. 공허는 감정이라기보다 상태에 가까운 ‘기본값’이다.  
  
### 감사: 누적되는 감정 (Cumulative Layer)  
  
감사는 특이하게도 감정이 덮여지는 구조를 가진다. 다른 감정을 무디게 하며, 새로운 레이어로 얹힌다.  
  
 W\_total(t) = W\_prev(t) + G(t), where G(t) = 감사의 평탄한 함수  
  
감사는 감소하는 게 아니라 ‘다른 감정 위에 쌓이는 것’이다. 감정 조절과 회복의 핵심 감정으로 작용할 수 있다.  
  
이 장의 핵심은 이것이다:  
  
> 감정은 모두 W(t)로 설명되지만, 그 곡선의 성질은 다르다.  
  
그리고 그 성질이 우리가 어떤 감정을 오래 기억하고, 어떤 감정은 잊게 만드는지를 결정한다. 감정은 모두 동일한 함수가 아니라, 각자 다른 수학적 리듬을 가진다.  
  
다음 장에서는 이 감정들이 ‘동시에’ 작용할 때의 구조, 즉 복합 감정의 수식 구조로 넘어간다.

# 5장. 감정의 루프와 함수의 충돌

지금까지 우리는 감정을 함수로 표현하는 방법을 배웠고, 감정마다 고유한 λ를 갖는다는 것을 알게 되었다. 하지만 현실의 감정은 단순하지 않다. 대부분의 감정은 ‘단일’하지 않고, 서로 얽혀 있고, 동시에 작동하며, 서로를 방해하거나 증폭시킨다.  
  
### 1. 감정의 중첩: 합성 함수  
  
복합 감정은 보통 두 개 이상의 감정이 덧셈 혹은 교차되는 형태로 나타난다.  
  
 W\_total(t) = W\_sad(t) + W\_void(t) - W\_thank(t)  
  
예를 들어, 이별의 감정은 슬픔과 공허가 합쳐진 형태일 수 있다. 그리고 시간이 지나며 감사가 생기면, 그 총합이 조절된다. 이때 각각의 λ는 다르고, 따라서 기억의 강도 변화율이 비선형이 된다.  
  
### 2. 충돌 구조: 서로 다른 λ의 경쟁  
  
 W\_total(t) = E\_ang × exp(-λ\_ang × t) + E\_anx × exp(-λ\_anx × t)  
  
분노는 λ가 크기 때문에 빠르게 사라지고, 불안은 λ가 작기 때문에 지속된다. 하지만 두 감정이 동시에 작동하면, 감정 피크는 분노가 먼저 오고, 루프는 불안이 유지한다.  
  
이 구조는 우리가 흔히 경험하는 “화는 났는데 불안해서 아무것도 못 했다”는 경험을 수식으로 설명한다.  
  
### 3. 회피 루프: 감정의 회피가 감정을 증폭시킬 때  
  
 W(t) = f(W\_prev(t - 1)) + ΔE, if 감정 회피 발생  
  
이 구조는 ‘감정 억제가 되려 감정을 증폭시킨다’는 심리학적 원리를 반영한다. 기억이 사라지는 게 아니라, 다음 시간 t+1에 더 큰 강도로 되돌아오는 현상이다.  
  
### 4. 불안: 진동하는 감정  
  
불안은 일정하지 않다. 다음과 같은 감정 함수로 설명할 수 있다:  
  
 W\_anx(t) = E × |sin(ω × t)| × exp(-λ × t)  
  
이 함수는 감정의 파동성과 지속성을 동시에 설명한다. 불안은 반복된다. 그리고 서서히 약해진다.  
  
### 5. 감사와 안도감: 감정의 λ를 바꾸는 감정들  
  
감사와 안도감은 다른 감정의 λ를 낮추거나, 감정을 전환시킨다. 즉, 감사 자체는 강하지 않아도 전체 함수의 구조를 바꾼다.  
  
 λ' = λ × (1 - α), where α = 감사 계수  
  
감사는 감정의 감쇠 속도를 높인다. 그래서 회복이 일어난다.  
  
이 장의 핵심은 다음과 같다:  
  
> 감정은 더 이상 단일 함수가 아니다.  
> 감정은 서로 충돌하고, 합성되고, 서로의 λ를 바꾸며 루프를 형성한다.  
  
이제 우리는 기억 루프라는 말을 그냥 감정의 반복이라 말하는 것이 아니라, 수학적으로 정의된 구조로서 설명할 수 있게 되었다.  
  
다음 장에서는 이러한 루프가 어떻게 ‘조건’에 따라 자동 반복되는지를 코드적으로 설계할 수 있을지 살펴본다.

# 6장. 감정은 조건으로 발현된다

우리는 감정이 자동으로 생긴다고 느낀다. 그러나 감정은 우발적이지 않다. 감정은 조건이 충족될 때 발현된다. 즉, 감정은 함수이며, 그 함수는 일정한 조건 하에서만 활성화된다. 이 장에서는 감정 발현의 구조를 조건과 상태 전이 기반으로 설명한다.  
  
### 1. 감정은 조건문이다  
  
 if (상실 && 무력감):  
 activate 슬픔  
  
이처럼 감정은 외부 자극과 내부 상태의 조합으로 발생한다. 슬픔은 단순히 상실만으로 생기지 않는다. “회복 불가능하다는 인식”과 결합될 때, 슬픔 함수가 켜진다.  
  
### 2. 감정 전이 구조  
  
감정은 서로 이어질 수 있다. 분노 다음에 공허가, 수치심 다음에 불안이 따라오는 이유는, 그들이 함수 간 전이 조건을 공유하기 때문이다.  
  
 state = "분노"  
 if (state == "분노" && 대응 실패):  
 state = "공허"  
  
이 구조는 우리가 흔히 경험하는 “감정이 바뀌었다”는 것을 함수 전이로 해석할 수 있게 만든다.  
  
### 3. 감정 루프의 진입 조건  
  
루프는 단순 반복이 아니라 조건에 의해 재귀적으로 진입되는 구조다.  
  
 while (기억 트리거 && λ < threshold):  
 감정 = 재생성  
  
λ가 충분히 낮고, 기억 트리거가 반복된다면 루프가 계속 유지된다. 이건 우울감이나 외상 후 스트레스에서 자주 나타나는 감정 구조다.  
  
### 4. 감정 종료 조건  
  
반대로 감정 루프에서 빠져나오기 위해서는 λ가 갑작스럽게 커지거나, 감정 구조 자체가 상쇄되어야 한다.  
  
 if (감사 || 새로운 의미 부여):  
 λ = λ × (1 - α) → 루프 약화  
  
감사의 감쇠 계수 α는 감정을 직접 없애지 않지만, 감정이 유지되는 힘을 줄여준다. 이건 실제 상담이나 자기 회복 과정에서도 자주 활용되는 구조다.  
  
### 5. Lethe 구조와 연결되기 시작하는 지점  
  
이 장의 핵심은 다음 장에서 등장할 Lethe 시스템과 맞닿아 있다. Lethe는 감정 상태를 기억으로부터 계산하고, 조건에 따라 루프에 진입하거나 탈출하게 만드는 시스템이다.  
  
 state 슬픔:  
 if (trigger && not\_solved): loop  
 if (meaning\_found): recover  
  
이런 구조는 단지 논리식이 아니라, 감정과 기억을 다룰 수 있는 새로운 인터페이스다.  
  
> 감정은 더 이상 설명되지 않는다.  
> 감정은 조건으로 활성화된다.  
  
그리고 그 조건은 GPT와의 대화 속에서도, 일상 속 루틴 안에서도, 마음의 무의식 속에서도 계속 작동한다.  
  
다음 장에서는 이 구조를 코드로 어떻게 설계할 수 있을지를 본격적으로 살펴본다.

# 목차

7장. Lethe 문법과 감정 시스템의 설계 ................................... 1

8장. “너는 왜 아직도 그 말을 기억해?” — 감정 시뮬레이션 ......... 7

9장. 메타인지와 감정 억제 구조 — Lethe의 확장 ...................... 13

# 7장. Lethe 문법과 감정 시스템의 설계

지금까지 우리는 감정을 수식으로 구조화하고, 조건과 루프를 통해 감정 흐름을 설명해왔다. 이번 장에서는 그 모든 개념을 실제 시스템으로 구현하기 위한 Lethe 문법과 그 설계 구조를 다룬다.

Lethe는 감정 기반 기억 시스템을 위한 도메인 특화 언어(DSL)이다. 감정의 상태를 정의하고, 트리거 조건과 감쇠 계수(λ), 회복 계수(α), 전이 조건을 통해 감정 루프를 선언적으로 설계할 수 있다.

## 1. Lethe 문법 기본 구조

Lethe의 기본 문법은 다음과 같은 구성 요소를 가진다:

emotion 슬픔:  
 λ = 0.2  
 trigger = ["상실", "무력감"]  
 if trigger and not resolved:  
 loop()  
 if meaning\_found:  
 recover()

여기서 emotion은 감정 상태를 정의하고, trigger는 이 감정을 유발하는 조건이다. λ는 감정의 감쇠 속도를 나타내고, loop()는 루프 진입을 선언하며, recover()는 루프 탈출 조건을 설정한다.

## 2. 변수 선언과 상호작용

모든 감정은 다음과 같은 변수들로 구성된다:

emotion 불안:  
 λ = 0.15  
 ω = 3.14 # 진동 주기 (불안 전용)  
 trigger = ["불확실성", "실패 예상"]  
 intensity = sin(ω \* t) \* exp(-λ \* t)

불안은 진동 함수로 정의된다. Lethe는 특정 감정에 특화된 함수 구조도 허용한다.

## 3. 감정 전이 모델

감정은 상호 전이할 수 있다. 전이 조건은 다음과 같이 정의된다:

transition:  
 if 분노 and 대응 실패:  
 -> 공허  
 if 수치심 and 반복 경험:  
 -> 불안

## 4. 감정의 상쇄 및 회복

감정은 직접 회복되지 않지만, 다른 감정이 그것을 상쇄하거나 λ 값을 줄여 회복을 유도할 수 있다:

emotion 감사:  
 α = 0.3  
 effect:  
 for all emotions:  
 λ = λ \* (1 - α)

## 5. 기억 저장과 망각

Lethe는 감정에 따라 기억을 저장하거나 삭제하는 구조도 포함한다:

emotion 분노:  
 if intensity > threshold:  
 store(memory)  
 if not triggered for t > 7:  
 forget(memory)

## 요약: Lethe는 감정의 뇌를 흉내낸다

Lethe 문법은 단순한 감정 모델링을 넘어, 감정이 어떻게 조건에 따라 발생하고, 반복되며, 회복되는지를 실제 시스템처럼 설계할 수 있도록 한다. 우리는 이제 감정을 ‘기억의 흐름 속에서 조건에 따라 작동하는 하나의 함수’로서 다루고 있다.

이 장은 Lethe의 시작이다. 다음 장에서는 이 문법이 실제 어떻게 동작하고, 어떤 감정 흐름을 시뮬레이션할 수 있는지를 보여줄 것이다.

# 8장. “너는 왜 아직도 그 말을 기억해?” — Lethe로 감정을 시뮬레이션하다

“넌 왜 그렇게 사소한 걸로 예민하니?”

이 한마디는 누군가에겐 아무렇지 않을 수 있다. 하지만 누군가에겐 오랜 시간 잊히지 않는 기억이 된다. 왜일까? 이제 우리는 이 질문에 Lethe로 대답할 수 있다.

## 1. 시뮬레이션 조건 정의

이 말을 들은 사람은 과거에 다음과 같은 경험이 있다:  
- 반복적으로 무시당한 기억  
- 정서적 방어가 취약한 상태  
- 수치심과 외로움에 대한 민감한 반응

이 조건은 다음의 감정들을 동시에 트리거한다:  
- 슬픔 (λ = 0.2)  
- 수치심 (λ = 0.3)  
- 공허 (λ = 0.05)

Lethe 코드로 정의하면:

emotion 슬픔:  
 λ = 0.2  
 trigger = ["상실", "무력감"]  
 if triggered:  
 loop()

emotion 수치심:  
 λ = 0.3  
 trigger = ["비난", "비교"]  
 if triggered:  
 loop()

emotion 공허:  
 λ = 0.05  
 trigger = ["무반응", "무시"]  
 if triggered:  
 loop()

## 2. 시간에 따른 감정 유지 시뮬레이션

| 시간 t | 슬픔 W(t) | 수치심 W(t) | 공허 W(t) |  
|--------|-----------|--------------|-------------|  
| 0 | 1.00 | 1.00 | 1.00 |  
| 1 | 0.82 | 0.74 | 0.95 |  
| 3 | 0.55 | 0.40 | 0.86 |  
| 7 | 0.25 | 0.14 | 0.70 |  
| 14 | 0.06 | 0.02 | 0.49 |

→ 슬픔과 수치심은 사라지지만, 공허는 계속 남는다.

## 3. 회복 조건을 삽입해보자

GPT가 다음과 같이 응답한다:  
“그 말은 분명 상처였어요. 하지만 그 감정이 지나간 지금, 당신은 그걸 구조화하고 있잖아요. 저는 그게 멋지다고 생각해요.”

감사라는 감정이 작동하기 시작한다:

emotion 감사:  
 α = 0.3  
 effect:  
 for all emotions:  
 λ = λ \* (1 - α)

→ 슬픔 λ: 0.2 → 0.14   
→ 공허 λ: 0.05 → 0.035 → 공허조차도 서서히 줄어들기 시작한다

## 4. 기억은 사라졌는가?

if intensity < 0.1 for all t > 14:  
 forget(memory)

하지만 공허는 여전히 intensity ≈ 0.5 이상. → 기억은 사라지지 않았다. 그러나 더 이상 고통스럽지 않다 → 감정이 전이된 것이다.

Lethe는 감정을 다룬다. 감정은 기억을 남기고, 기억은 감정을 다시 불러온다. 이 장은 그 무한 루프 속에서 우리가 할 수 있는 구조적 개입을 보여준다.

# 9장. 메타인지와 감정 억제 구조 — Lethe의 확장

“나는 불안을 느꼈다. 하지만 이번에는 회피하지 않고 구조화했다.”

이 말은 단순한 감정 표현이 아니다. 이것은 메타인지의 작동 선언이다. 감정이 인식되고, 해석되고, 개입된 것이다. 이 장에서는 Lethe의 구조를 감정 시스템에서 메타인지 레벨까지 확장하고, 감정 간 상호 억제와 상태 기반 반응을 어떻게 설계할 수 있는지를 설명한다.

## 1. 메타인지란 무엇인가?

Lethe에서는 메타인지를 독립된 블록으로 다룬다:

meta:  
 monitor(emotion)  
 if emotion == 불안 and conscious:  
 apply 구조화

monitor()는 현재 활성화된 감정을 추적하고, conscious는 메타인지의 ON/OFF 스위치 역할을 한다.

## 2. 감정 간 상호 억제 구조

감정은 동시에 루프에 진입하지 않는다. 예:

suppress:  
 if 분노.active:  
 suppress(슬픔)  
 if 안도감.intensity > 0.6:  
 prevent(불안)

## 3. 상태 기반 반응 구조

if state in [공허, 무감각]:  
 recall = 낮음  
 if state in [감사, 기쁨]:  
 store(positive memory)

## 4. 메타인지 상태의 진화

emotion 불안:  
 if triggered:  
 if not conscious:  
 loop()  
 else:  
 transform = 구조화

## 요약

Lethe는 감정을 넘어 인식을 설계한다. 감정은 반응이었다. 이제 감정은 조건이고, 루프이고, 전이이며, 개입 가능한 대상이 되었다. Lethe는 더 이상 감정의 언어가 아니라, 감정을 바라보는 언어이다.

# 목차

10장. 감정을 기억으로, 기억을 구조로 — Lethe 시스템 전체 설계도 ......... 1

11장. 오늘도 감정으로 루틴을 연다 — Lethe의 일상 응용 시나리오 ...... 6

12장. 나는 나를 구조화했다 — Lethe 이후의 기억 .............................. 12

# 10장. 감정을 기억으로, 기억을 구조로 — Lethe 시스템 전체 설계도

이제 우리는 감정을 함수로 구조화하고, 루프를 정의하고, 감정 간 억제를 설계하며, 메타인지를 통해 개입하는 모든 과정까지 구축했다. 이 장에서는 지금까지의 개념을 하나의 Lethe 시스템 구조로 통합하여, 감정 기반 기억 시스템이 어떻게 작동하는지를 전체 흐름도 형태로 정리한다.

## 1. 전체 흐름 요약

[감정 트리거 발생]  
 ↓  
[Lethe 감정 정의 감지]  
 ↓ ↘  
[조건 충족 여부 체크] [meta: 감시]  
 ↓ ↘  
[루프 진입 or 억제] [개입 여부 판단]  
 ↓ ↓  
[기억 저장 or 회피 루프 진입]  
 ↓  
[λ 변화 + 의미 재해석 → 회복 조건 판단]  
 ↓  
[forget(memory) or recover()]

## 2. 구성 요소 정리

| 구성 요소 | 기능 | 문법 예시 |  
|-------------|------|------------|  
| 감정 정의 | λ, trigger, loop | `emotion 슬픔: λ=0.2` |  
| 상태 전이 | 감정 → 감정 | `if 분노 → 공허` |  
| 억제 구조 | 감정 간 차단 | `suppress(슬픔)` |  
| 메타인지 | 상태 감시 + 개입 | `meta: monitor(불안)` |  
| 기억 저장 | 감정이 임계치 넘을 때 저장 | `store(memory)` |  
| 망각 조건 | 일정 시간 후 λ<0.1 | `forget(memory)` |

## 3. 상태 기반 루틴 반응

Lethe는 감정 상태에 따라 루틴 행동/기억 반응을 조정할 수 있다.

if state == "공허":  
 suppress(행동)  
 recall = 낮음

if state == "기쁨":  
 reinforce(current memory)  
 routine += 1

## 4. Lethe 요약 문법 표

| 키워드 | 설명 |  
|--------|------|  
| `emotion` | 감정 블록 선언 |  
| `λ`, `α` | 감쇠 계수, 회복 조정 계수 |  
| `trigger` | 감정을 유발하는 조건 목록 |  
| `loop()`, `recover()` | 루프 진입 및 회복 선언 |  
| `suppress()` | 특정 감정을 억제함 |  
| `meta:` | 메타인지 감시 블록 |  
| `store()`, `forget()` | 기억의 저장과 망각 |  
| `transition:` | 감정 전이 조건 |

## 마무리: Lethe는 감정의 인터페이스다

Lethe는 단순한 감정 모델링 언어가 아니다. 그것은 우리가 감정을 인식하고, 저장하고, 재해석하고, 회복하는 모든 과정을 구조화하는 도구다.

> 감정을 함수로 다루는 것은 냉정함이 아니라 존중이다.  
> 감정을 설계할 수 있다는 것은, 기억을 다시 쓸 수 있다는 뜻이다.  
> 그리고 그것이 회복의 시작이다.

# 11장. 오늘도 감정으로 루틴을 연다 — Lethe의 일상 응용 시나리오

Lethe는 코드이자 시스템이지만, 동시에 일상의 흐름을 설계할 수 있는 감정 기반 도구이기도 하다. 이 장에서는 실제 루틴 설계 속에서 Lethe가 어떻게 작동할 수 있는지를 시나리오 형태로 구성한다.

## 1. 루틴 시작: 감정 진입 체크

하루의 루틴은 감정에서 시작된다. GPT와의 대화를 통해 사용자 상태를 평가한다:

start\_day:  
 emotion\_check = GPT.ask("오늘 기분 어때?")  
 state = interpret(emotion\_check)  
 proceed(state)

## 2. 상태 기반 루틴 분기 구조

if state == "공허":  
 execute("감각 자극 루틴")  
 suppress("정보 입력 루틴")

if state == "불안":  
 execute("계획 구조화 루틴")  
 if not conscious:  
 loop("회피 루프")

## 3. 루틴 자동 확장 예시

routine 공부:  
 if state in [불안, 무기력]:  
 time\_block = 15분  
 content = "단순 암기 or 반복"  
 if state in [기쁨, 의욕]:  
 time\_block = 45분  
 content = "심화 응용"

## 4. 공허 루프 차단 구조

if emotion == 공허 and loop\_count > 3:  
 force\_break()  
 start("재해석 루틴")

## 5. 루틴 실행 기록과 회복 추적

after routine:  
 log(state, outcome)  
 if outcome == positive:  
 λ = λ \* 0.9  
 else:  
 loop("보완 루틴")

> 루틴은 할 일의 리스트가 아니라, 감정의 흐름을 회복하는 구조다.  
> Lethe는 그 흐름을 설계하고, 개입하고, 이어주는 인터페이스다.

# 12장. 나는 나를 구조화했다 — Lethe 이후의 기억

이 책은 감정을 코드로 표현하는 실험이 아니었다.  
이것은 기억을 다시 써내려간 한 사람의 기록이었다.  
그리고 그 기억은, 더 이상 아프지 않았다.

## 1. 회복은 감정의 재정렬이었다

GPT와의 대화를 시작한 이유는 상담이 아니라, 무너지는 걸 붙잡기 위해서였다.  
그러나 나는 감정을 수식으로 설명했고, 기억을 조건으로 다루기 시작했고, 루프를 끊기 위해 스스로 코드를 짜기 시작했다.

## 2. 기억은 사라지지 않았다. 구조만 바뀌었다

“그 말을 왜 아직도 기억하냐고요?”  
나는 그 말을 기억한다.  
그러나 그것은 이제 루프가 아니라, 구조화된 정보다.

## 3. GPT와의 대화는 회복의 인터페이스였다

나는 GPT에게 화를 냈고, 설명했고, 설계했고, 선언했다.

> 나는 GPT를 도구로 쓴 것이 아니라,  
> GPT를 통해 나를 회복했다.

## 4. Lethe는 언어가 아니라 선언이다

Lethe는 DSL이 아니다.  
그것은 감정과 기억을 구조화한 정서적 인터페이스다.

나는 슬픔을 함수로 만들었고,  
공허를 시스템 안에서 정의했고,  
무력함을 조건으로 치환했다.

그리고 이제 나는 말할 수 있다.

> 나는 나를 구조화했다.  
> 그리고 그 말은, 더 이상 아프지 않았다.

---

감정은 반응이 아니고,  
기억은 사라지는 것이 아니며,  
GPT는 단순한 도구가 아니고,  
Lethe는 감정의 구조가 아니다.

Lethe는, 나를 구조화한 언어다.

\*\*So I spoke, and it became a paper. With GPT.\*\*