

활동 증빙서류

목차

파이썬, HTML 코딩 공부

[1-1 파이선](#)

[1-2 HTML, CSS](#)

2. 유니티 프로젝트

[2-1 PROJECT 1 WALKING MECHANISM](#)

[2-2 LONG PROJECT 2 SWORD REINFORCE](#)

[2-3 PROJECT 3 BINARY NUMBER QUIZ](#)

[2-4 PROJECT 4 CALCULATOR](#)

3. 그래픽 작업

[3-1 그래픽 작업을 시작 한 계기](#)

[3-2 포스터](#)

[3-3 로고](#)

[3-4 느낀 점](#)

4. 간단 게임 기획

[4-1 간단 기획서](#)

[4-2 느낀 점](#)

1-1 파일

간단한 연산 활용

나눗셈

```
/*
나눗셈 계산기 만들기
두 수를 외부에서 입력받아
(출력)
나눗셈의 몫은 __이고 나머지는 __이다.
// %
첫번째수 num1
두번째수 num2 변수를
*/
```

```
print("나누어질 수 입력")
num1 = float(input())
print("나눌 수 입력")
num2 = float(input())
result = num1 // num2
remain = num1 % num2
print("나눗셈의 몫은", result, "이고 나머지는", remain, "입니다.")
```

나누어질 수 입력
45
나눌 수 입력
6
나눗셈의 몫은 7.0이고 나머지는 3.0입니다.

섭시 온도 변환

```
/*
섭시온도
풀이어는 절은 0도, 풀는 절은 100도로 기준하고 그 사이간격을 100으로 나눈 몫을 몰할
(32F) (212F)
*/
섭시온도 = (섭시온도 * 1.8) + 32
```

섭시온도를 환경온도로 변환하는 프로그램을 작성하시오.
섭시온도는 외부에서 입력받습니다.

```
celsius : 섭시온도 변수
fahrenheit : 환경온도 변수
...
print("변환할 섭시온도 입력") # 안내문구
celsius = float(input()) # 외부에서 값을 받아서 문자 -> 숫자로 바꿔 저장
fahrenheit = (celsius * 1.8) + 32 # 환경온도로 변경해서 저장
print('섭시온도:', celsius) # 출력
print('환경온도:', fahrenheit) # 출력
```

변환할 섭시온도 입력
32
섭시온도 : 32.0
환경온도 : 89.6

BMI 지수

```
/*
비만지수 BMI 구하기
BMI = 몸무게/(키*키)**2
(변수)
이름 name
키 height
몸무게 weight
...
(name) name 끝의 키는 cm이고 몸무개는 kg입니다.
BMI 지수는 __입니다.
round(bmi, 2)
...
print("이름을 입력하시오")
name = input()
print("키를 입력하시오")
height = float(input())
print("몸무게를 입력하시오")
weight = float(input())
print(name, "님의 키는", height, "cm이고 몸무개는", weight, "kg입니다.")
BMI = weight / (height * 100)**2
print("BMI 지수는", BMI, "입니다.")
*/
```

이름을 입력하시오
^^^
키를 입력하시오
180
몸무게를 입력하시오
70
^^^ 님의 키는 180.0 cm이고 몸무개는 70.0 kg입니다.
BMI 지수는 21.604938271604937입니다.

반복문을 이용한 활용

```
4 # 1. 특정한 단 * 수 출력
5 print('단')
6 dan = (input())
7 print(dan, '단')
8
9 for i in range(1,10) :
10     result = dan * i
11     print(dan,'*',i,"=",result)
12
13 print("*****")
14
15 # 2. 1단 ~ 9단까지 전체 출력
16 for i in range(1,10) :
17     for j in range(1,10) :
18
19         result = i * j
20         print(i, "*",j,"=",result)
21         print() # 줄 띄움
22
23 # 3. 1~9단에서 수가 짝수인것만 출력
24 for i in range(1,10) : # 단
25     for j in range(1,10) : # 수
26
27         if j%2 == 0 : # 짝수
28             result = i*j # 계산
29             print(i,'*',j,'=',result) # 출력
30
31 # 4. 1~9단 중에서 단이 짝수 단인것만 출력
32 for i in range(1,10) :
33     if i%2 == 0 : # 짝수
34         for j in range(1,10) : # 수 증가
35             result = i * j # 계산
36             print(i,'*',j,'=',result) # 출력
37
38 # 5. while문으로 구구단 전체를 출력해보자
39 i = 1 # 단
40 j = 1 # 수
41
42 while i < 10 : # 단
43     while j < 10 : # 수
44         result = i * j
45         print(i,'*',j,'=',result) # i * j = 결과값
46         j = j + 1
47
48 i = i + 1
49 j = 1
```

4 * 8 = 32	3 * 6 = 18	6 * 2 = 12	5 * 2 = 10	6 * 8 = 48
4 * 9 = 36	3 * 7 = 21	6 * 3 = 18	5 * 3 = 15	6 * 9 = 54
5 * 1 = 5	1 * 2 = 2	6 * 4 = 24	5 * 4 = 20	8 * 1 = 8
5 * 2 = 10	3 * 8 = 24	8 * 7 = 56	5 * 5 = 25	8 * 2 = 16
5 * 3 = 15	1 * 3 = 3	3 * 9 = 27	5 * 6 = 30	5 * 8 = 40
5 * 4 = 20	1 * 4 = 4	6 * 5 = 30	6 * 7 = 42	8 * 3 = 24
5 * 5 = 25	1 * 5 = 5	8 * 8 = 64	6 * 8 = 48	8 * 4 = 32
5 * 6 = 30	1 * 6 = 6	7 * 9 = 63	6 * 9 = 54	6 * 5 = 30
5 * 7 = 35	1 * 7 = 7	9 * 4 = 36	7 * 10 = 70	8 * 6 = 48
5 * 8 = 40	1 * 8 = 8	4 * 5 = 20	9 * 5 = 45	8 * 7 = 56
5 * 9 = 45	1 * 9 = 9	6 * 6 = 36	9 * 6 = 54	7 * 8 = 56
6 * 1 = 6	1 * 10 = 10	8 * 9 = 72	7 * 9 = 63	7 * 9 = 63
6 * 2 = 12	2 * 2 = 4	6 * 7 = 42	9 * 1 = 9	7 * 10 = 70
6 * 3 = 18	2 * 3 = 6	6 * 8 = 48	9 * 2 = 18	8 * 1 = 8
6 * 4 = 24	2 * 4 = 8	6 * 9 = 54	9 * 3 = 27	8 * 2 = 16
6 * 5 = 30	2 * 5 = 10	8 * 10 = 80	9 * 4 = 36	1 * 3 = 3
6 * 6 = 36	2 * 6 = 12	7 * 1 = 7	9 * 5 = 45	8 * 3 = 24
6 * 7 = 42	2 * 7 = 14	9 * 2 = 18	9 * 6 = 54	1 * 4 = 4
6 * 8 = 48	2 * 8 = 16	6 * 3 = 18	9 * 7 = 63	1 * 5 = 5
6 * 9 = 54	2 * 9 = 18	8 * 4 = 32	9 * 8 = 72	1 * 6 = 6
7 * 1 = 7	2 * 10 = 20	7 * 5 = 35	9 * 9 = 81	1 * 7 = 7
7 * 2 = 14	2 * 11 = 22	7 * 6 = 42	7 * 10 = 70	1 * 8 = 8
7 * 3 = 21	2 * 12 = 24	7 * 7 = 49	9 * 8 = 72	1 * 9 = 9
7 * 4 = 28	2 * 13 = 26	7 * 8 = 56	9 * 9 = 81	1 * 10 = 10
7 * 5 = 35	2 * 14 = 28	7 * 9 = 63	9 * 10 = 90	1 * 11 = 11
7 * 6 = 42	2 * 15 = 30	7 * 10 = 70	9 * 11 = 99	1 * 12 = 12
7 * 7 = 49	2 * 16 = 32	7 * 11 = 77	9 * 12 = 108	1 * 13 = 13
7 * 8 = 56	2 * 17 = 34	7 * 12 = 84	9 * 13 = 117	1 * 14 = 14
7 * 9 = 63	2 * 18 = 36	7 * 13 = 91	9 * 14 = 126	1 * 15 = 15
7 * 10 = 70	2 * 19 = 38	7 * 14 = 98	9 * 15 = 135	1 * 16 = 16
7 * 11 = 77	2 * 20 = 40	7 * 15 = 105	9 * 16 = 144	1 * 17 = 17
7 * 12 = 84	2 * 21 = 42	7 * 16 = 112	9 * 17 = 153	1 * 18 = 18
7 * 13 = 91	2 * 22 = 44	7 * 17 = 120	9 * 18 = 162	1 * 19 = 19
7 * 14 = 98	2 * 23 = 46	7 * 18 = 128	9 * 19 = 171	1 * 20 = 20
7 * 15 = 105	2 * 24 = 48	7 * 19 = 135	9 * 20 = 180	1 * 21 = 21
7 * 16 = 112	2 * 25 = 50	7 * 20 = 142	9 * 21 = 189	1 * 22 = 22
7 * 17 = 120	2 * 26 = 52	7 * 21 = 150	9 * 22 = 198	1 * 23 = 23
7 * 18 = 128	2 * 27 = 54	7 * 22 = 158	9 * 23 = 207	1 * 24 = 24
7 * 19 = 135	2 * 28 = 56	7 * 23 = 166	9 * 24 = 216	1 * 25 = 25
7 * 20 = 142	2 * 29 = 58	7 * 24 = 174	9 * 25 = 225	1 * 26 = 26
7 * 21 = 150	2 * 30 = 60	7 * 25 = 182	9 * 26 = 234	1 * 27 = 27
7 * 22 = 158	2 * 31 = 62	7 * 26 = 190	9 * 27 = 243	1 * 28 = 28
7 * 23 = 166	2 * 32 = 64	7 * 27 = 198	9 * 28 = 252	1 * 29 = 29
7 * 24 = 174	2 * 33 = 66	7 * 28 = 206	9 * 29 = 261	1 * 30 = 30
7 * 25 = 182	2 * 34 = 68	7 * 29 = 214	9 * 30 = 270	1 * 31 = 31
7 * 26 = 190	2 * 35 = 70	7 * 30 = 222	9 * 31 = 279	1 * 32 = 32
7 * 27 = 198	2 * 36 = 72	7 * 31 = 230	9 * 32 = 288	1 * 33 = 33
7 * 28 = 206	2 * 37 = 74	7 * 32 = 238	9 * 33 = 297	1 * 34 = 34
7 * 29 = 214	2 * 38 = 76	7 * 33 = 246	9 * 34 = 306	1 * 35 = 35
7 * 30 = 222	2 * 39 = 78	7 * 34 = 254	9 * 35 = 315	1 * 36 = 36
7 * 31 = 230	2 * 40 = 80	7 * 35 = 262	9 * 36 = 324	1 * 37 = 37
7 * 32 = 238	2 * 41 = 82	7 * 36 = 270	9 * 37 = 333	1 * 38 = 38
7 * 33 = 246	2 * 42 = 84	7 * 37 = 278	9 * 38 = 342	1 * 39 = 39
7 * 34 = 254	2 * 43 = 86	7 * 38 = 286	9 * 39 = 351	1 * 40 = 40
7 * 35 = 262	2 * 44 = 88	7 * 39 = 294	9 * 40 = 360	1 * 41 = 41
7 * 36 = 270	2 * 45 = 90	7 * 40 = 302	9 * 41 = 369	1 * 42 = 42
7 * 37 = 278	2 * 46 = 92	7 * 41 = 310	9 * 42 = 378	1 * 43 = 43
7 * 38 = 286	2 * 47 = 94	7 * 42 = 318	9 * 43 = 387	1 * 44 = 44
7 * 39 = 294	2 * 48 = 96	7 * 43 = 326	9 * 44 = 396	1 * 45 = 45
7 * 40 = 302	2 * 49 = 98	7 * 44 = 334	9 * 45 = 405	1 * 46 = 46
7 * 41 = 310	2 * 50 = 100	7 * 45 = 342	9 * 46 = 414	1 * 47 = 47
7 * 42 = 318	2 * 51 = 102	7 * 46 = 350	9 * 47 = 423	1 * 48 = 48
7 * 43 = 326	2 * 52 = 104	7 * 47 = 358	9 * 48 = 432	1 * 49 = 49
7 * 44 = 334	2 * 53 = 106	7 * 48 = 366	9 * 49 = 441	1 * 50 = 50
7 * 45 = 342	2 * 54 = 108	7 * 49 = 374	9 * 50 = 450	1 * 51 = 51
7 * 46 = 350	2 * 55 = 110	7 * 50 = 382	9 * 51 = 459	1 * 52 = 52
7 * 47 = 358	2 * 56 = 112	7 * 51 = 390	9 * 52 = 468	1 * 53 = 53
7 * 48 = 366	2 * 57 = 114	7 * 52 = 398	9 * 53 = 477	1 * 54 = 54
7 * 49 = 374	2 * 58 = 116	7 * 53 = 406	9 * 54 = 486	1 * 55 = 55
7 * 50 = 382	2 * 59 = 118	7 * 54 = 414	9 * 55 = 495	1 * 56 = 56
7 * 51 = 390	2 * 60 = 120	7 * 55 = 422	9 * 56 = 504	1 * 57 = 57
7 * 52 = 398	2 * 61 = 122	7 * 56 = 430	9 * 57 = 513	1 * 58 = 58
7 * 53 = 406	2 * 62 = 124	7 * 57 = 438	9 * 58 = 522	1 * 59 = 59
7 * 54 = 414	2 * 63 = 126	7 * 58 = 446	9 * 59 = 531	1 * 60 = 60
7 * 55 = 422	2 * 64 = 128	7 * 59 = 454	9 * 60 = 540	1 * 61 = 61
7 * 56 = 430	2 * 65 = 130	7 * 60 = 462	9 * 61 = 549	1 * 62 = 62
7 * 57 = 438	2 * 66 = 132	7 * 61 = 470	9 * 62 = 558	1 * 63 = 63
7 * 58 = 446	2 * 67 = 134	7 * 62 = 478	9 * 63 = 567	1 * 64 = 64
7 * 59 = 454	2 * 68 = 136	7 * 63 = 486	9 * 64 = 576	1 * 65 = 65
7 * 60 = 462	2 * 69 = 138	7 * 64 = 494	9 * 65 = 585	1 * 66 = 66
7 * 61 = 470	2 * 70 = 140	7 * 65 = 502	9 * 66 = 594	1 * 67 = 67
7 * 62 = 478	2 * 71 = 142	7 * 66 = 510	9 * 67 = 603	1 * 68 = 68
7 * 63 = 486	2 * 72 = 144	7 * 67 = 518	9 * 68 = 612	1 * 69 = 69
7 * 64 = 494	2 * 73 = 146	7 * 68 = 526	9 * 69 = 621	1 * 70 = 70
7 * 65 = 502	2 * 74 = 148	7 * 69 = 534	9 * 70 = 630	1 * 71 = 71
7 * 66 = 510	2 * 75 = 150	7 * 70 = 542	9 * 71 = 639	1 * 72 = 72
7 * 67 = 518	2 * 76 = 152	7 * 71 = 550	9 * 72 = 648	1 * 73 = 73
7 * 68 = 526	2 * 77 = 154	7 * 72 = 558	9 * 73 = 657	1 * 74 = 74
7 * 69 = 534	2 * 78 = 156	7 * 73 = 566	9 * 74 = 666	1 * 75 = 75
7 * 70 = 542	2 * 79 = 158	7 * 74 = 574	9 * 75 = 675	1 * 76 = 76
7 * 71 = 550	2 * 80 = 160	7 * 75 = 582	9 * 76 = 684	1 * 77 = 77
7 * 72 = 558	2 * 81 = 162	7 * 76 = 590	9 * 77 = 693	1 * 78 = 78
7 * 73 = 566	2 * 82 = 164	7 * 77 = 598	9 * 78 = 702	1 * 79 = 79
7 * 74 = 574	2 * 83 = 166	7 * 78 = 606	9 * 79 = 711	1 * 80 = 80
7 * 75 = 582	2 * 84 = 168	7 * 79 = 614	9 * 80 = 720	1 * 81 = 81

함수

```
1 ...
2 컴퓨터 구조를 이해해야 하는 부분으로 S/W와 H/W가 어떻게 동작하는지를
3 이해해야 한다.
4
5 1. 할수과함식 = 파이썬은 두 가지 방식을 혼합 사용한다.
6
7 1) 값에 의한 호출(call by value)
8     - 할수에 인수를 넣을 때 값만 넘김
9     - 할수 안의 인수값 변경시, 호출된 변수에 영향을 주지 않음.
10
11 2) 참조로함 (call by reference)
12     - 할수에 인수를 넣을 때 메모리 주소를 넘김
13     - 할수 안의 인수값 변경시, 호출된 변수값도 변경됨.
14
15 2. 예)
16     5 <-- x = 5
17     /
18     / ----- y = x 이렇게 하면 x와 y는 주소를 공유해서
19     y도 x가 가르키는 주소를 참조하게 된다.
20     그러므로 y를 출력하면 5가 출력된다.
21 ...
22 # 변수의 초기화
23 def f(x):
24
25     y = x
26     x = 5
27
28     return y*x
29
30 # 위의 할수를 호출 실행해보자
31 a = 3
32 print(f(a)) # f(x) 할수의 매개변수 x로 3을 넘겨주어서 f() 함수에서 계산된 값을 반환 출력할 것이다
33 print(a) # f(x) 할수 밖의 x를 출력할 것이다.
34
35 # 변수의 값저장 및 객체생성
36
37 def spam(eggs):
38
39     eggs.append(1)
40     eggs=[2,3]
41
42     print('eggs[1]',eggs)
```

9
3
eggs[] [2, 3]
ham[]: [0, 1]
10
10
20
10

딕셔너리

```
1 ...
2     - key, value를 한쌍으로 저장하는 자료구조
3     - 풀용
4         . 일락 : key, value
5         . 풀학 : key로 풀학함
6     - 키는 중복되면 안됨, 만들어가지는 않지만 찾는 값에서 오류 발생
7     - {}(중괄호)를 사용
8     - 풀법
9         (설명) 딕셔너리 변수 = {키1:값1, 키2:값2, 키3:값3, ...}
10        (풀법) 딕셔너리 변수[키]
11
12     - 값-> 리스트, 투플, 세트, 딕셔너리 자기 자신 모두 가능
13 ...
14 # 딕셔너리 생성1 - 키를 중복시켜서 어떤 현상이 발생하는지 (키는 중복되면 안됨)
15 student1 = {20150203: '공통', 20150203: '공통', 20150203: '이순신'}
16 print(student1[20150203]) # 마지막 것이 출력되고 이렇게 되면 찾는 값을 대로 못찾음
17
18 # 딕셔너리 생성2
19 student2 = {20150203: '공통', 20150204: '공통', 20150205: '이순신', 20150206: '강감찬'}
20 # 딕셔너리 값을 출력
21 print(student2[20150203])
22 print(student2[20150204])
23 print(student2[20150205])
24 print(student2[20150206])
25
26 # 위에 출력문을 for문을 이용해 출력
27 for i in range(len(student2)): # len(student2) = 들어가있는 값의 크기
28
29     k = int('2015020' + str(i+3)) # 형변환을 하지 않으면 오류
30     print("for문을 사용한 출력", student2[k])
```

```
(1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
(1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
튜플의 크기는 9개
임
튜플 출력 설악 금강 한라
튜플에서 mount[-1]*3 연산 관악관악관악
('설악', '금강', '한라', '백두', '관악')
```

(느낀점)

처음에 한국에서 파이선이 코딩 입문용 언어로 많이 쓰이고 있다는 것을 전부터 알고 있었는데 한번 공부해 보고 싶었습니다. 이렇게 기회가 찾아와서 기쁜 마음으로 하려고 했는데 직접 해보니 코딩 방식이 조금 부자연스러운 듯한 느낌이 들었습니다. 그래서 파이선을 학생들에게 가르쳐주시는 선생님께 파이선이 한국에서 특히 공부 용도로 많이 사용되는 것 같은데 왜 그런지에 대해 이유를 물어봤습니다. 그런데 파이선이 외국에서 해킹과 프로그래밍 등의 정보통신 분야에서 많이 쓰여서 한국도 파이선으로 공부한다는 것 이었습니다. 처음에는 파이선이 교육용 그 이상도 이하도 아닌 줄만 알았는데 굉장히 의외였습니다. 하지만 C와 C#을 배우고 온 저로서는 더 쓰기 좋고 직관적인 언어가 많은 것 같은데 외국 분들이 파이선을 많이 쓴다는 것은 조금 의아했습니다. 이상한 부분도 있었지만 현재 많이 쓰이고 있는 언어라고 해서 도움이 될 것 같았고 또한 한국 교육과정에서 많이 가르치고 있는 이유는 무엇인지 알아보면 좋을 것 같아 공부해 보았습니다. C언어를 미리 공부하고 파이선을 공부한 것이라 문법적인 차이점만 제외하면 크게 다른 부분은 없었습니다. 하지만 C언어와 번갈아 가면서 하다 보니 파이선을 계속 공부하면서 세미콜론과 괄호의 부재가 허전하게 느껴졌지만 공부해 보면서 점차 적응되기 시작하였고 계획했던 공부를 모두 끝마치게 되었습니다. 파이선을 지속해서 공부해 보면서 보이지 않던 장점들이 조금 보이기 시작했습니다. 첫 번째로 편했던 점은 딕셔너리 같은 선언하기 복잡한 방식을 구현하는 데 큰 힘이 들지 않는 것었습니다. C#에서는 따로 지정해 주어야 했는데 그런 과정이 필요 없어서 좋았습니다. 두 번째로 편했던 점은 매우 간결하고 쉽게 코드 작성이 가능하다는 점이었습니다. 장점이 드러났지만 역시 간결해져 버리니 직관적이지 않고 세세한 부분을 만들기에는 조금 무리가 있어 보였습니다. 하지만 파이선에는 아직 배우지 않은 함수나 문법들이 많아 보였는데 그것들까지 모두 활용하면 하나의 팬찮은 프로그램을 만들 수도 있다는 생각이 들었습니다. 이번 파이선 공부는 옛날에 만들어진 C언어 같은 도구와 대비되게 현재 만들어지는 새로운 프로그래밍 도구를 체험하게 해주었고 현재 그리고 과거부터 추구해온 프로그래밍의 간결함이 어떤 건지 확실히 알게 되는 시간이었던 것 같습니다. 앞으로 이런 새로운 언어와 환경에도 적응해야 겠다는 생각이 확실히 들었습니다.

1-2 HTML

기본태그

```

1 <!--
2 . 마크업
3   - 파일 내에 삽입되는 일련의 문자 또는 기호 (다른 용어로 태그)
4   . 논리적 구조 정의
5   . 둘서가 프린터로 출력되거나 화면으로 표시될 때 어떠한 형태로 보여질 것인지 지시하는 역할
6
7 - 마크업 언어
8 . SGML : 국제표준 마크업 언어
9   - 매우 강력한 표현력
10  - 문법과 구성이 복잡
11  - 군사, 우주, 항공 등의 특수분야에서 사용
12
13 . HTML : 웹에서 정보를 표현할 목적으로 만든 마크업 언어
14   - SGML 대 그중 일부를 추출해서 만든 형태
15   - 정해진 태그만 사용이 가능
16   - 구조화 어려움
17
18 . XML : SGML로 부터 필요한 기능만 뽑아서 새롭게 정의한 마크업 언어
19   - 마크업간의 관계를 정의할 수 있고 논리적 구조정의가 쉽다.
20   - 입식의 태그를 사용하지 직접 만들어서 사용할 수 있다.
21
22 HTML 기본
23   . 주석 : <!-- (내용) --> 처리
24   . 태그는 대소문자 구분 인감 (HTML = html)
25   . 파일 : *.html로 되어 있어야 웹상에서 보여짐
26   . 문서의 기본 골격
27
28 <html>
29   <head>
30     <meta>           : 정보
31     <title>(문서의 제목)</title> : 제목
32   </head>
33   <body>
34     (문서 내용)
35   </body>
36 </html>
37
38 -->
39
40 <!DOCTYPE html>
41 <html>
42   <head>
43     <meta charset="UTF-8">
44     <title>Insert title here</title>
45   </head>
46   <body>
47     화면 디자인
48   </body>
49 </html>

```

화면 디자인

태그 정리

```

1 <!--
2
3 -->
4
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <meta charset="utf-8">
10    <title>Insert title here</title>
11  </head>
12  <body>
13
14    <p> 문서의 기본 바탕색은 흰색이고 기본 글자색은 검정색 <p>
15
16    . topmargin = "수치" : 상단 여백 지정 <br>
17    . leftmargin = "수치" : 왼쪽 여백 지정 <br>
18    . rightmargin = "수치" : 오른쪽 여백 지정 <br>
19    . bottommargin = "수치" : 아래 여백 지정 <br>
20    . bgcolor = "색상" : 배경색 지정 <br>
21    . background = "주소" : 배경색으로 사용될 이미지 지정 <br>
22    . text = "색상" : 글자색 지정 <br>
23    . link = "색상" : 하이퍼링크로 설정된 글자의 색을 지정 <br>
24    . alink = "색상" : 링크된 문자를 클릭 할 때 변화되는 색을 지정 <br>
25    . vlink = "색상" : 링클로 사용한 후의 문자색 지정 <br>
26
27 <!--*
28   p태그 : 줄바꿈, 간격이 넓음
29   br태그 : 줄바꿈, 간격이 좁음, 시작 태그가 없음
30 -->
31 </body>
32 </html>

```

문서의 기본 바탕색은 흰색이고 기본 글자색은 검정색

```

topmargin = "수치" : 상단 여백 지정
leftmargin = "수치" : 왼쪽 여백 지정
rightmargin = "수치" : 오른쪽 여백 지정
bottommargin = "수치" : 아래 여백 지정
bgcolor = "색상" : 배경색 지정
background = "주소" : 배경색으로 사용될 이미지 지정
text = "색상" : 글자색 지정
link = "색상" : 하이퍼링크로 설정된 글자의 색을 지정
alink = "색상" : 링크된 문자를 클릭 할 때 변화되는 색을 지정
vlink = "색상" : 링클로 사용한 후의 문자색 지정

```

링크

```

1 <!--
2   하이퍼링크를 삽입할 때는 <a> anchor태그를 사용함
3
4   <a>의 속성을
5     . href="경로" : 링크를 통해 이용하고자 하는 곳을 경로/주소 지정
6     . name="이름" : 문서의 특정 위치를 이름으로 지정
7     . target="창의 위치" : 링크를 클릭 하였을 때 문서가 출력될 프레임 지정
8     . title="설명" : 링크에 대한 설명 표시
9     _blank : 링크된 내용을 새로운 창으로 표시
10    _self : 링크된 내용을 현재의 창에 표시
11    _top : 현재의 창이 프레임으로 구성된 경우 프레임이 사라지고 하나의 화면에 링크된 내용 표시
12
13    -parent : 링크 부분이 있는 프레임에 새로운 프레임이 만들어져 링크된 내용 표시
14    프레임명 : 링크된 내용을 표시할 프레임 이름을 직접 지정하는 경우
15    . title="설명내용" : 링크에 대한 설명 표시
16 -->
17
18 <!DOCTYPE html>
19 <html lang = "kor">
20   <head>
21     <meta charset="utf-8">
22     <title>하이퍼링크 삽입 - href</title>
23   </head>
24   <body>
25     <a href="http://www.naver.com">네이버</a> <br>
26     <a href="http://youtube.com" target="_blank">유튜브</a> <br>
27     <a href="http://www.daum.net" target="_blank">
28       다음</a> <br>
29   </body>
30 </html>

```



(느낀점)

웹디자인도 프로그래밍을 배우면서 관심이 조금 있었는데 곁으로 듣기에는 원래 하던 프로그래밍과는 아주 다르다고 들어서 궁금증으로 공부하기 시작했습니다. 첫인상은 예상했던 것처럼 내가 알던 프로그래밍이랑은 다르다는 것이었습니다. 더욱더 깊게 들어갈수록 내가 기존에 해왔던 프로그래밍과는 다른 것 같다는 생각이 들었습니다. 전부를 공부한 것이 아니므로 이런 생각이 들게 된 건지도 모르겠지만 이 언어는 프로그래밍보다는 디자인에 더욱 치중되어 있다는 것이 와닿았습니다. 그 때문에 프로그래밍적 사고까지는 필요하지 않은 쉬운 언어인 것처럼 보였습니다. 최근 웹디자인 분야는 개인이 만들기 쉽고 이미 저는 최근 웹사이트를 어떻게 운영하고 개설하는지 알고 싶어서 워드프레스 호스팅 비용을 내고 자신만의 웹사이트를 만들어본 경력도 있습니다. (<https://vrowdicing.com/>) 매우 적은 비용만 지급해도 서식, 운영 관련 정보, 각종 애드온까지 쉽게 웹디자인이 가능한 좋은 시스템이 갖추어져 있었습니다. 웹디자인을 쉽게 할 수 있는 현실이기는 하지만 html도 쓰일 날이 생길 수 있으므로 한 번 공부해 보는 것도 팬찮으리라 생각합니다. 또 이런 방식으로 구성돼있다는 것에 신기함을 느꼈습니다. 그리고 태그 사용이 주가 되는 언어인 것 같은데 하나하나 모두 찾아서 외우시면서 웹을 만드신 분들이 존경스러웠습니다. 웹상에서 쓰는 또 다른 언어인 자바스크립트는 활용되는 분야가 html보다 조금 넓은 것 같아 자바스크립트를 한 번 공부해 보고 싶다는 생각도 들었습니다.

2-1 PROJECT 1 WALKING

MECHANISM

(1) 이 게임을 만들게 된 계기

유니티에 처음 입문하고 얼마 되지 않아서 만든 프로그램이다. 유니티 안에서는 키 입력을 horizontal과 vertical을 통해 임의의 코드를 작성하기만 하면 자동으로 키 입력을 받아서 대상을 움직입니다. 하지만 저는 if문이 프로그래밍에서 상당히 많이 쓰는 문법인 것을 알아냈기 때문에 관련 학습을 조금 더 하고 싶었습니다. 그래서 키 입력을 자동으로 받는 것이 아닌 if문과 input, 그리고 유니티 안에 존재하는 이동 함수인 transform.Translate를 이용하여 구성했습니다. 또한 이번 프로젝트는 처음으로 임의의 함수를 사용한 프로젝트이기도 합니다.

(2) 기본적인 게임 설명

왼쪽 이동, 오른쪽 이동, 앞으로 이동, 뒤로 이동, 대각선 이동 점프가 가능합니다.

(3) 기본 화면



중앙에 존재하는 캡슐이 움직이는 주체이고 rotation 값이 고정되어 넘어지지 않습니다.

(4) 베이스 코드

1번 이미지

```
70  //<summary>
71  //<summary> 이동을 위한 키 입력 및 전송
72  //</summary>
73  //참조 0개
74  private void Update()
75  {
76      if (Input.GetKeyDown(KeyCode.W))
77      {
78          transform.Translate(Vector3.forward * mMovePower * Time.deltaTime);
79          InputKey_W = true;
80          Left_Up();
81          Right_Up();
82          //Debug.Log("W 누름");
83      }
84      else if (Input.GetKeyUp(KeyCode.W))
85      {
86          InputKey_W = false;
87          //Debug.Log("W 뗄");
88      }
89
90      if (Input.GetKeyDown(KeyCode.S))
91      {
92          transform.Translate(Vector3.back * mMovePower * Time.deltaTime);
93          InputKey_S = true;
94          Left_Down();
95          Right_Down();
96          //Debug.Log("S 누름");
97      }
98      else if (Input.GetKeyUp(KeyCode.S))
```

2번 이미지

```
131     if (Input.GetKeyDown(KeyCode.Space))
132     {
133         if (mJumpFlag == true)
134         {
135             rb.AddForce(transform.up * mJumpPower);
136         }
137     }
```

3번 이미지

```
139     void Left_Up()
140     {
141         if (InputKey_W == true && InputKey_A == true)
142         {
143             transform.Translate(new Vector3(-mMovePower, 0, mMovePower) * mMovePower * Time.deltaTime);
144             //Debug.Log("대각선 이동 성공적");
145         }
146         if (InputKey_W == false)
147         {
148             transform.Translate(Vector3.right * mMovePower * Time.deltaTime);
149         }
150         if (InputKey_A == false)
151         {
152             transform.Translate(Vector3.forward * mMovePower * Time.deltaTime);
153         }
154     }
155
156     void Right_Up()
157     {
158         if (InputKey_W == true && InputKey_D == true)
159         {
160             transform.Translate(new Vector3(mMovePower, 0, mMovePower) * mMovePower * Time.deltaTime);
161             //Debug.Log("대각선 이동 성공적");
162         }
163         if (InputKey_W == false)
164         {
165             transform.Translate(Vector3.right * mMovePower * Time.deltaTime);
166         }
167     }
```

1번 이미지

Update 함수를 통해 input으로 항상 입력을 받을 수 있게 설정했습니다. 입력을 받게 되면 transform.Translate의 Vector3의 forward 값을 통해 앞으로 나가는 힘을 직접 캡슐에 가하게 됩니다.

2번 이미지

스페이스 버튼을 누를 시 Addforce를 사용하여 위를 향해 직접적으로 물리적인 힘을 주어 점프가 가능합니다.

3번 이미지

대각선 처리를 하기 위해 InputKey_W bool 변수를 추가해 키 입력을 받았는지 받지 않았는지 확인시킵니다. 각 bool의 확인을 통해 함수를 통해 대각선 이동 처리를 해주게 됩니다.

(5) 문제점

문제1. 함수의 사용

원래는 if문으로만 구성하려고 했으나 대각선 이동 때문에 if문에서 직접적으로 참조하게 되는 함수가 필요할 것으로 생각되어 다른 프로그래밍에서 쓰던 함수를 유니티에서 옮겨와서 사용했기 때문에 처음에는 원래 쓴 함수와 다를 것이라 예상해 쓰기를 주저했지만 코딩에서 함수는 중요한 부분이라고 생각되었기 때문에 찾아서 사용하는데 시간이 조금 걸렸지만 함수의 사용이라는 매우 큰 부분을 학습하게 되어 의미 있는 시간이였던 것 같습니다.

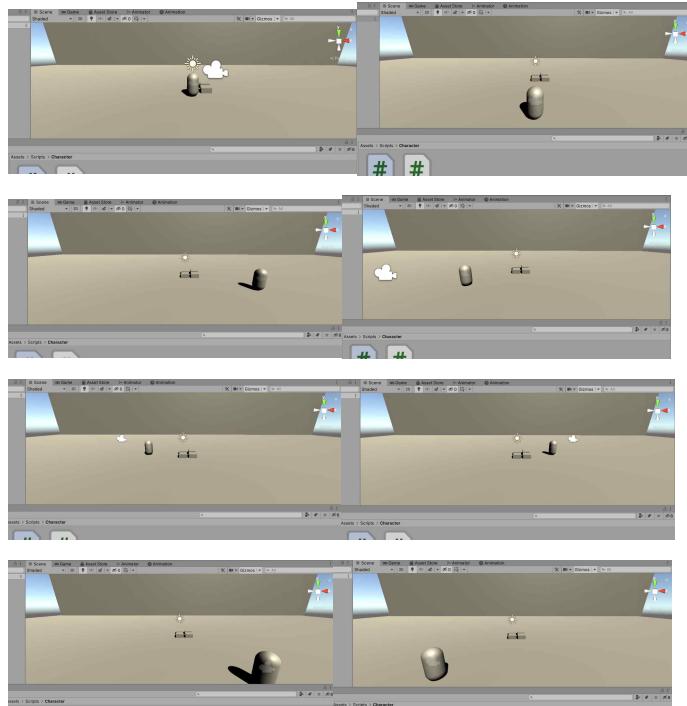
문제2. 대각선 이동1 (베이스코드 143, 159번째 줄 참조)

함수의 사용 말고도 앞, 뒤, 옆 이동은 쉬운 편이었으나 대각선 이동 자체를 생각해 내는 것도 큰 고비중에 하나였는데 예를 들면 vector3의 forward값과 left값을 같이 넣어주면 대각선 방향으로 이동할 것으로 생각했으나 11시 방향으로 이동하게 되어 당황스러웠던 기억이 납니다. 그때는 유니티에 대해 잘 아는 편이 아니었기 때문에 엔진에서 주어진 값을 이용해서 if문을 통해 대각선을 구현하는 방법은 여려울 것 같았습니다. 그래서 x, y, z축으로 작용하는 힘을 직접 주기로 해서 대각선 이동을 성공적으로 구현했습니다.

문제3. 대각선 이동2 (베이스코드 145 ~ 152번째 줄 참조)

대각선을 구현하고도 키를 누르고 있을 때 키가 중복되어 이상한 방향으로 가기도 했는데 이 부분도 많은 고민을 했습니다. 그러던 중 bool 변수를 이용해 누르고 있다면 true 키에서 손을 뗐다면 false로 지정해 이동에 필요한 bool값들이 모두 true가 아니면 이동을 하지 않게 제한을 해 주었습니다.

(6) 구현화면



순서대로 위, 아래, 오른쪽, 왼쪽, 왼쪽 위, 오른쪽 위, 왼쪽 아래, 오른쪽 아래 이동

(7) 이 프로젝트를 진행하면서 느끼고 배운 점

이 프로젝트는 전부터 계속 느꼈던 유니티의 장점을 느낄 수 있었던 나 자신이 무슨 코드를 짜고 움직여 보면서 성취감을 제대로 느낄 수 있는 프로젝트였습니다. 그리고 기존에 쓰던 if문에서 벗어나 다양한 문법을 스스로 찾아서 사용해보는 앞으로의 활동에 영향을 끼칠 중요한 프로젝트였던 것 같습니다.

2-2 LONG PROJECT 2

SWORD REINFORCE

(1) 이 게임을 만들게 된 계기

직접적으로 만든 계기는 기존 한국의 과금을 매우 많이 해야하는 RPG게임과는 다른 확률이 투명하게 공개되어있으면서 무과금으로 간편하고 재미있게 플레이 가능한 확률게임을 만들고 싶었습니다. 이러한 이유로 다른 프로젝트들은 게임도 만들고 코딩을 공부할 겸 만든 프로젝트들이 대다수지만 이번 프로젝트는 게임 프로그래밍을 배우기 이전에 순전히 만들고 싶었던 게임을 만들어보고 싶다는 생각으로 시작하게 되었습니다. 또한 UI게임을 만들어보고 싶었는데 보기보다 만들기 어렵고 흔히 말하는 노가다를 통한 코딩이 주를 이룬다고 하여 게임을 만드는데 노가다가 들어가면 얼마나 힘들지 UI를 통해 경험해 보고 싶었습니다.

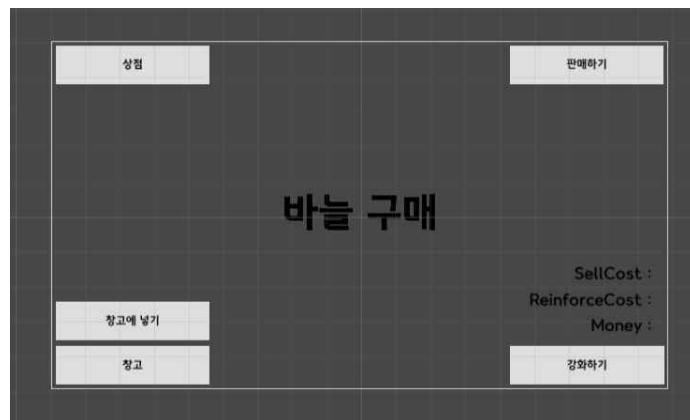
(2) 기본적인 게임 설명

검을 강화하여 저장하고 판매하면서 돈을 많이 버는 게임입니다.

더 높은 강화 횟수를 가진 검을 판매할수록 돈이 많이
늘어납니다.

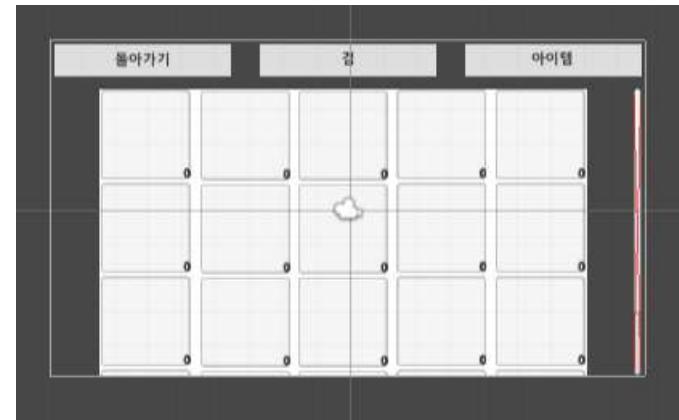
(3) 기본 화면

‘메인게임’ 화면



‘상점’, ‘창고’ 버튼을 누를 시 상점화면으로 이동합니다
‘판매하기’를 누를 시 검이 판매됩니다. ‘창고에 넣기’ 버튼을 누를 시 검이 창고에 들어갑니다.
‘강화하기’를 누르면 검이 강화됩니다.

‘창고’ 화면



돌아가기 버튼을 누르면 메인게임 화면으로 돌아갑니다.
‘검’버튼을 누르면 검 스크롤창으로
이동합니다.’아이템’버튼을 누르면 아이템 스크롤창으로
이동합니다. 스크롤창의 버튼에 검이나 아이템이 존재할
경우 이미지와 그 밑에 0이라고 되어있는 숫자칸에
개수가 표시되며 만약 존재할 시 판매나 사용이
가능합니다.(아이템은 아직 구현하지 못했습니다.)

(4) 베이스 코드

'강화' 버튼 스크립트

1번 이미지

2번 이미지

```
72
73     public void Click()
74     {
75         switch (ReinforceCount)
76         {
77             case 0:
78                 ReinforceCost.text = "강화비용 : " + 150 + 0.3;
79                 SelCost.text = "비교비용 : " + 150;
80                 M1.LstMoney -= 150 + 0.3;
81                 break;
82
83             case 1:
84                 ReinforceCost.text = "강화비용 : " + 250 + 0.3;
85                 SelCost.text = "비교비용 : " + 250;
86                 M1.LstMoney -= 250 + 0.3;
87                 break;
88
89             case 2:
90                 ReinforceCost.text = "강화비용 : " + 400 + 0.3;
91                 SelCost.text = "비교비용 : " + 400;
92                 M1.LstMoney -= 400 + 0.3;
93                 break;
94
95             case 3:
96                 ReinforceCost.text = "강화비용 : " + 700 + 0.3;
97                 SelCost.text = "비교비용 : " + 700;
98                 M1.LstMoney -= 400 + 0.3;
99                 break;
100
101            case 4:
102                ReinforceCost.text = "강화비용 : " + 1500 + 0.3;
103                SelCost.text = "비교비용 : " + 1500;
104                M1.LstMoney -= 700 + 0.3;
105                break;
106
107            case 5:
108                ReinforceCost.text = "강화비용 : " + 5000 + 0.3;
109                SelCost.text = "비교비용 : " + 5000;
110                M1.LstMoney -= 1500 + 0.3;
111                break;
112
113            case 6:
114                ReinforceCost.text = "강화비용 : " + 15000 + 0.3;
115                SelCost.text = "비교비용 : " + 15000;
116                M1.LstMoney -= 5000 + 0.3;
117                break;
118
119            case 7:
120                ReinforceCost.text = "강화비용 : " + 50000 + 0.3;
121                SelCost.text = "비교비용 : " + 50000;
122                M1.LstMoney -= 15000 + 0.3;
123                break;
124
125            case 8:
126                ReinforceCost.text = "강화비용 : " + 250000 + 0.3;
127                SelCost.text = "비교비용 : " + 250000;
128                M1.LstMoney -= 50000 + 0.3;
129                break;
130
131            case 9:
132                ReinforceCost.text = "강화비용 : " + 1000000 + 0.3;
133                SelCost.text = "비교비용 : " + 1000000;
134                M1.LstMoney -= 250000 + 0.3;
135                break;
136
137        }
138    }
139
140    if (ReinforceCount == 1)
141    {
142        Target = Gameobject.Find("BuyMiddle(Clone)");
143        Destroy(Target);
144    }
145
146    ReinforcePreset = Random.Range(0.0f, 100.0f); // 0/100 확률 변수
147
148    if (ReinforcePercent <= RPDef) // 0/100 확률 변수
149    {
150        Target = Gameobject.Inc("BuyMiddle(Clone)");
151        Destroy(Target);
152        Destroy(Target);
153        ReinforceCount += 1;
154        Gameobject.Incp("Transform", position =
155        Debug.Log("1"));
156        return;
157    }
158
159    else
160    {
161        ReinforceCount = 0;
162        SelCost.text = "비교비용 : 0";
163        ReinforceCost.text = "강화비용 : 0";
164        if (ReinforceCount < 0)
165        {
166            ReinforceCount = 0;
167        }
168    }
169
170    if (ReinforceCount == 1)
171    {
172        if (ReinforcePercent <= RPDef)
173        {
174            Target = Gameobject.Find("Rt(Clone)");
175            Destroy(Target);
176            Instantiate(R2);
177            ReinforceCount += 1;
178            Debug.Log("1");
179        }
180
181        else
182        {
183            Target = Gameobject.Find("Rt(Clone)");
184            Destroy(Target);
185            Instantiate(RMiddle);
186            ReinforceCount = 0;
187            SelCost.text = "비교비용 : 0";
188            ReinforceCost.text = "강화비용 : 0";
189            if (ReinforceCount < 0)
190            {
191                ReinforceCount = 0;
192            }
193        }
194    }
195 }
```

1번 이미지

클릭할 때마다 switch case문을 통해 강화비용과 판매비용을 바꾸며 돈을 감소시킵니다.

2번 이미지

검을 강화한 횟수에 따라 확률을 바꿔주고 강화에 성공하면 다음 검 이미지로 강화에 실패하면 타겟 설정을 통해 강화하던 검을 삭제합니다.

'창고에 넣기' 버튼 스크립트

1번 이미지

2번 이미지

```

196 ///////////////
197 ///////////////
198 ///////////////
199 ///////////////
200 ///////////////
201 ///////////////
202 ///////////////
203 ///////////////
204 ///////////////
205 ///////////////
206 ///////////////
207 ///////////////
208 ///////////////
209 ///////////////
210 ///////////////
211 ///////////////
212 ///////////////
213 ///////////////
214 ///////////////
215 ///////////////
216 ///////////////
217 ///////////////
218 ///////////////
219 ///////////////
220 ///////////////
221 ///////////////
222 ///////////////
223 ///////////////
224 ///////////////
225 ///////////////
226 ///////////////
227 ///////////////
228 ///////////////
229 ///////////////
230 ///////////////
231 ///////////////
232 ///////////////
233 ///////////////
234 ///////////////
235 ///////////////
236 ///////////////
237 ///////////////
238 ///////////////
239 ///////////////
240 ///////////////
241 ///////////////
242 ///////////////
243 ///////////////
244 ///////////////
245 ///////////////
246 ///////////////
247 ///////////////
248 ///////////////
249 ///////////////
250 ///////////////
251 ///////////////
252 ///////////////
253 ///////////////
254 ///////////////
255 ///////////////
256 ///////////////
257 ///////////////
258 ///////////////
259 ///////////////
260 ///////////////
261 ///////////////
262 ///////////////
263 ///////////////
264 ///////////////
265 ///////////////
266 ///////////////
267 ///////////////
268 ///////////////
269 ///////////////
270 ///////////////
271 ///////////////
272 ///////////////
273 ///////////////
274 ///////////////
275 ///////////////
276 ///////////////
277 ///////////////
278 ///////////////
279 ///////////////
280 ///////////////
281 ///////////////
282 ///////////////
283 ///////////////
284 ///////////////
285 ///////////////
286 ///////////////
287 ///////////////
288 ///////////////
289 ///////////////
290 ///////////////
291 ///////////////
292 ///////////////
293 ///////////////
294 ///////////////
295 ///////////////
296 ///////////////
297 ///////////////
298 ///////////////
299 ///////////////
300 ///////////////
301 ///////////////
302 ///////////////
303 ///////////////
304 ///////////////
305 ///////////////
306 ///////////////
307 ///////////////
308 ///////////////
309 ///////////////
310 ///////////////
311 ///////////////
312 ///////////////
313 ///////////////
314 ///////////////
315 ///////////////
316 ///////////////
317 ///////////////
318 ///////////////
319 ///////////////
320 ///////////////
321 ///////////////
322 ///////////////
323 ///////////////
324 ///////////////
325 ///////////////
326 ///////////////
327 ///////////////
328 ///////////////
329 ///////////////
330 ///////////////
331 ///////////////
332 ///////////////
333 ///////////////
334 ///////////////
335 ///////////////
336 ///////////////
337 ///////////////
338 ///////////////
339 ///////////////
340 ///////////////
341 ///////////////
342 ///////////////
343 ///////////////
344 ///////////////
345 ///////////////
346 ///////////////
347 ///////////////
348 ///////////////
349 ///////////////
350 ///////////////
351 ///////////////
352 ///////////////
353 ///////////////
354 ///////////////
355 ///////////////
356 ///////////////
357 ///////////////
358 ///////////////
359 ///////////////
360 ///////////////
361 ///////////////
362 ///////////////
363 ///////////////
364 ///////////////
365 ///////////////
366 ///////////////
367 ///////////////
368 ///////////////
369 ///////////////
370 ///////////////
371 ///////////////
372 ///////////////
373 ///////////////
374 ///////////////
375 ///////////////
376 ///////////////
377 ///////////////
378 ///////////////
379 ///////////////
380 ///////////////
381 ///////////////
382 ///////////////
383 ///////////////
384 ///////////////
385 ///////////////
386 ///////////////
387 ///////////////
388 ///////////////
389 ///////////////
390 ///////////////
391 ///////////////
392 ///////////////
393 ///////////////
394 ///////////////
395 ///////////////
396 ///////////////
397 ///////////////
398 ///////////////
399 ///////////////
400 ///////////////
401 ///////////////
402 ///////////////
403 ///////////////
404 ///////////////
405 ///////////////
406 ///////////////
407 ///////////////
408 ///////////////
409 ///////////////
410 ///////////////
411 ///////////////
412 ///////////////
413 ///////////////
414 ///////////////
415 ///////////////
416 ///////////////
417 ///////////////
418 ///////////////
419 ///////////////
420 ///////////////
421 ///////////////
422 ///////////////
423 ///////////////
424 ///////////////
425 ///////////////
426 ///////////////
427 ///////////////
428 ///////////////
429 ///////////////
430 ///////////////
431 ///////////////
432 ///////////////
433 ///////////////
434 ///////////////
435 ///////////////
436 ///////////////
437 ///////////////
438 ///////////////
439 ///////////////
440 ///////////////
441 ///////////////
442 ///////////////
443 ///////////////
444 ///////////////
445 ///////////////
446 ///////////////
447 ///////////////
448 ///////////////
449 ///////////////
450 ///////////////
451 ///////////////
452 ///////////////
453 ///////////////
454 ///////////////
455 ///////////////
456 ///////////////
457 ///////////////
458 ///////////////
459 ///////////////
460 ///////////////
461 ///////////////
462 ///////////////
463 ///////////////
464 ///////////////
465 ///////////////
466 ///////////////
467 ///////////////
468 ///////////////
469 ///////////////
470 ///////////////
471 ///////////////
472 ///////////////
473 ///////////////
474 ///////////////
475 ///////////////
476 ///////////////
477 ///////////////
478 ///////////////
479 ///////////////
480 ///////////////
481 ///////////////
482 ///////////////
483 ///////////////
484 ///////////////
485 ///////////////
486 ///////////////
487 ///////////////
488 ///////////////
489 ///////////////
490 ///////////////
491 ///////////////
492 ///////////////
493 ///////////////
494 ///////////////
495 ///////////////
496 ///////////////
497 ///////////////
498 ///////////////
499 ///////////////
500 ///////////////
501 ///////////////
502 ///////////////
503 ///////////////
504 ///////////////
505 ///////////////
506 ///////////////
507 ///////////////
508 ///////////////
509 ///////////////
510 ///////////////
511 ///////////////
512 ///////////////
513 ///////////////
514 ///////////////
515 ///////////////
516 ///////////////
517 ///////////////
518 ///////////////
519 ///////////////
520 ///////////////
521 ///////////////
522 ///////////////
523 ///////////////
524 ///////////////
525 ///////////////
526 ///////////////
527 ///////////////
528 ///////////////
529 ///////////////
530 ///////////////
531 ///////////////
532 ///////////////
533 ///////////////
534 ///////////////
535 ///////////////
536 ///////////////
537 ///////////////
538 ///////////////
539 ///////////////
540 ///////////////
541 ///////////////
542 ///////////////
543 ///////////////
544 ///////////////
545 ///////////////
546 ///////////////
547 ///////////////
548 ///////////////
549 ///////////////
550 ///////////////
551 ///////////////
552 ///////////////
553 ///////////////
554 ///////////////
555 ///////////////
556 ///////////////
557 ///////////////
558 ///////////////
559 ///////////////
560 ///////////////
561 ///////////////
562 ///////////////
563 ///////////////
564 ///////////////
565 ///////////////
566 ///////////////
567 ///////////////
568 ///////////////
569 ///////////////
570 ///////////////
571 ///////////////
572 ///////////////
573 ///////////////
574 ///////////////
575 ///////////////
576 ///////////////
577 ///////////////
578 ///////////////
579 ///////////////
580 ///////////////
581 ///////////////
582 ///////////////
583 ///////////////
584 ///////////////
585 ///////////////
586 ///////////////
587 ///////////////
588 ///////////////
589 ///////////////
590 ///////////////
591 ///////////////
592 ///////////////
593 ///////////////
594 ///////////////
595 ///////////////
596 ///////////////
597 ///////////////
598 ///////////////
599 ///////////////
600 ///////////////
601 ///////////////
602 ///////////////
603 ///////////////
604 ///////////////
605 ///////////////
606 ///////////////
607 ///////////////
608 ///////////////
609 ///////////////
610 ///////////////
611 ///////////////
612 ///////////////
613 ///////////////
614 ///////////////
615 ///////////////
616 ///////////////
617 ///////////////
618 ///////////////
619 ///////////////
620 ///////////////
621 ///////////////
622 ///////////////
623 ///////////////
624 ///////////////
625 ///////////////
626 ///////////////
627 ///////////////
628 ///////////////
629 ///////////////
630 ///////////////
631 ///////////////
632 ///////////////
633 ///////////////
634 ///////////////
635 ///////////////
636 ///////////////
637 ///////////////
638 ///////////////
639 ///////////////
640 ///////////////
641 ///////////////
642 ///////////////
643 ///////////////
644 ///////////////
645 ///////////////
646 ///////////////
647 ///////////////
648 ///////////////
649 ///////////////
650 ///////////////
651 ///////////////
652 ///////////////
653 ///////////////
654 ///////////////
655 ///////////////
656 ///////////////
657 ///////////////
658 ///////////////
659 ///////////////
660 ///////////////
661 ///////////////
662 ///////////////
663 ///////////////
664 ///////////////
665 ///////////////
666 ///////////////
667 ///////////////
668 ///////////////
669 ///////////////
670 ///////////////
671 ///////////////
672 ///////////////
673 ///////////////
674 ///////////////
675 ///////////////
676 ///////////////
677 ///////////////
678 ///////////////
679 ///////////////
680 ///////////////
681 ///////////////
682 ///////////////
683 ///////////////
684 ///////////////
685 ///////////////
686 ///////////////
687 ///////////////
688 ///////////////
689 ///////////////
690 ///////////////
691 ///////////////
692 ///////////////
693 ///////////////
694 ///////////////
695 ///////////////
696 ///////////////
697 ///////////////
698 ///////////////
699 ///////////////
700 ///////////////
701 ///////////////
702 ///////////////
703 ///////////////
704 ///////////////
705 ///////////////
706 ///////////////
707 ///////////////
708 ///////////////
709 ///////////////
710 ///////////////
711 ///////////////
712 ///////////////
713 ///////////////
714 ///////////////
715 ///////////////
716 ///////////////
717 ///////////////
718 ///////////////
719 ///////////////
720 ///////////////
721 ///////////////
722 ///////////////
723 ///////////////
724 ///////////////
725 ///////////////
726 ///////////////
727 ///////////////
728 ///////////////
729 ///////////////
730 ///////////////
731 ///////////////
732 ///////////////
733 ///////////////
734 ///////////////
735 ///////////////
736 ///////////////
737 ///////////////
738 ///////////////
739 ///////////////
740 ///////////////
741 ///////////////
742 ///////////////
743 ///////////////>

```

1번 이미지

창고 안의 이미지를 바꾸고 넣은 개수를 추가해 준 다음 강화비용과 판매비용을 초기화합니다.

2번 이미지

검 태그를 이용해서 검 태그를 가진 오브젝트를 타겟으로 설정해 제거하고 강화 횟수를 초기화합니다.

'판매' 버튼 스크립트

1번 이미지

2번 이미지

```

1 ///////////////
2 ///////////////
3 ///////////////
4 ///////////////
5 ///////////////
6 ///////////////
7 ///////////////
8 ///////////////
9 ///////////////
10 ///////////////
11 ///////////////
12 ///////////////
13 ///////////////
14 ///////////////
15 ///////////////
16 ///////////////
17 ///////////////
18 ///////////////
19 ///////////////
20 ///////////////
21 ///////////////
22 ///////////////
23 ///////////////
24 ///////////////
25 ///////////////
26 ///////////////
27 ///////////////
28 ///////////////
29 ///////////////
30 ///////////////
31 ///////////////
32 ///////////////
33 ///////////////
34 ///////////////
35 ///////////////
36 ///////////////
37 ///////////////
38 ///////////////
39 ///////////////
40 ///////////////
41 ///////////////
42 ///////////////
43 ///////////////
44 ///////////////
45 ///////////////
46 ///////////////
47 ///////////////
48 ///////////////
49 ///////////////
50 ///////////////
51 ///////////////
52 ///////////////
53 ///////////////
54 ///////////////
55 ///////////////
56 ///////////////
57 ///////////////
58 ///////////////
59 ///////////////>

```

1번 이미지

클릭할 때마다 switch case문을 통해 강화비용과 판매비용을 초기화하고 돈을 추가합니다.

2번 이미지

'창고에 넣기' 버튼처럼 검 태그를 이용해서 검 태그를 가진 오브젝트를 타겟으로 설정해 제거하고 강화 횟수를 초기화합니다.

(5) 문제점

문제1. 데이터 선언의 문제

원래는 데이터를 선언해 주는 새로운 방식을 찾아서 적용하려고 하였으나 도저히 찾을 수가 없어서 스크립트마다 해당하는 프리팹(하이어락기 뷰에서 뿐만 아니라 따로 파일로 저장)을 지정하여 하나하나 넣는 방식으로 진행하였습니다. 이를 데이터를 선언하는 방식으로 했다면 더욱 간결한 코드가 될 수 있었을 것 같아 아쉬웠습니다. 하지만 추후 한번 더 도전해 볼 의향이 있습니다.

문제2. 오브젝트 삭제 문제

Destroy를 통해 Target으로 지정된 오브젝트를 삭제하고 BuyNiddle을 instantiate로 생성하는 코드에 전반적인 문제가 생겼습니다. 이를 해결하려고 구글링을 하면서 며칠을 계속 고민했습니다. 여러 가지 경우의 수중 잘 되는 경우의 수도 발생했습니다. 저는 이때 큰 깨달음을 얻었습니다. 프리팹이 아니라 하이어락기뷰에서 BuyNiddle을 끌어오고 있었습니다. 쉽게 풀어서 말하자면 Destroy를 통해 한번 사라지면 끝인 하이어락기뷰에서 존재했던 BuyNiddle을 삭제해 버렸으니 instantiate는 하이어락기뷰에서 존재하고 있는 BuyNiddle을 복제하라고 하니 성립 될 수가 없었습니다. 이 문제는 간단하게 프리팹의 BuyNiddle로 모두 교체해 주면서 해결이 되었습니다.

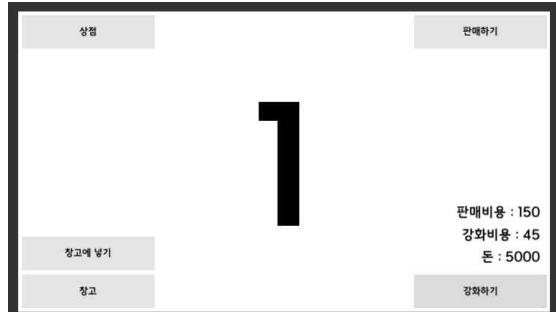
문제3. 혼자 진행하는 프로젝트의 단점

이 프로젝트는 혼자 길게 진행하는 프로젝트이기 때문에 확실히 같이 하는 프로젝트 보다 힘이 듦다는 느낌을 더 받았습니다. 그리고 문제2번과 같은 간단하고 나중에 보면 어이없는 오류도 다른 사람의 시각으로 보아야 한다는 것은 충분히 알고 있지만 현실은 그게 쉽지 않아 나 자신의 시각만으로 오류를 해결해야 하게 때문에 시간이 상당히 많이 걸리는 편이었습니다.

(6) 구현 화면



'강화하기' 클릭



‘강화하기’ 클릭 (강화성공)



‘판매하기’ 클릭



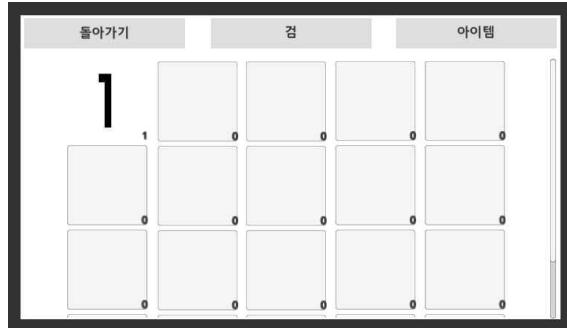
‘강화하기’ 클릭



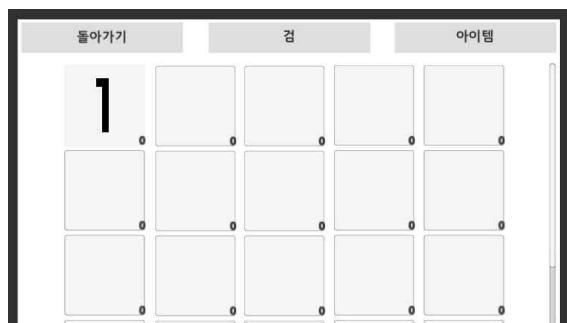
창고에 넣기’ 클릭



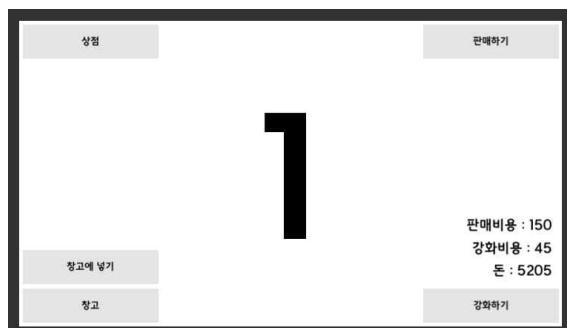
‘창고’ 클릭



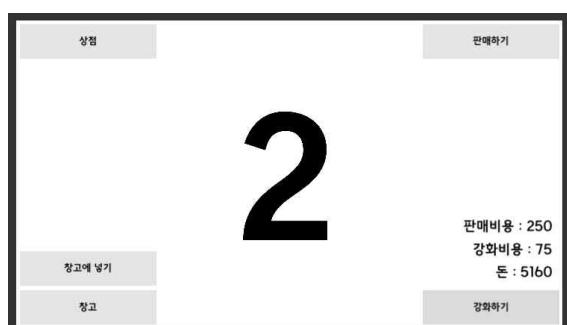
검이 존재하는 버튼 클릭(한번 강화된 검에 1개로 표시된 버튼 클릭)



‘돌아가기’ 클릭



‘강화하기’ 클릭



‘강화하기’ 클릭 (강화성공)



(강화실패)

(7) 프로젝트를 구성하며 느끼고 배운 점

UI라는 것이 눈에 띄게 보이는 프로그래밍은 아니면서 흔히 막노동이라고 불리는 이유가 있었음을 깨달았습니다. 그리고 프로그래머들은 UI 프로그래밍처럼 조금은 무의미한 것처럼 보이는 코딩을 해야 할 때도 있다고 생각하게 되었습니다. 이런 게임 프로젝트를 시작하면서 그리고 다른 비슷한 프로그래밍도 마찬가지고 별로 눈에 보이지 않는 일을 해야 할 때가 있다고 말입니다. 하지만 이렇게 보이지 않은 뒤에서 잘 보이지 않는 프로그래밍을 하는 사람이 더욱 자기 일에 열중할수록 결국 프로그램의 기동이자 근본이 단단해지게 된다는 것을 저는 알게 되었고 프로그래머도 종류가 참 많은데 어떤 프로그래밍을 하던 모두 필요한 사람들이라는 것을 알게 되었습니다. 그리고 처음에는 1인 개발을 아이디어만 존재하면 어렵지 않을 것 같았는데 이번에 경험을 통해 1인 개발자를 바라보면 분업화가 없는 1인 개발은 전혀 쉽지 않은 일이란 것을 알게 되었습니다. 쉬운 일은 없다는 말을 상당히 많이 들었는데 이번에 경험해 보면서 맞는 말인 것 같다는 생각도 들었습니다. 또한 프로그래밍도 개발하면서 알아보며 UI 프로그래밍을 별로 좋지 않은 시각으로 보는 사람들도 많은데 저는 UI도 매력이 있는 것 같다는 생각이 들었습니다. 비록 즉각적으로 자기가 한 코딩이 뭔가를 만들었다는 느낌을 주는 것이 아닌 나중에 모든 버튼과 스크롤 같은 UI가 톱니바퀴처럼 맞물려 맞아떨어지듯 만들고 났을 때의 쾌감이 참 좋았던 것 같습니다. 앞으로도 진행될 프로젝트이기 때문에 특히 많은 깨달음을 주는 큰 프로젝트가 될 것 같습니다. 현재는 아직 매우 미숙하지만 지금까지 그래왔듯이 본 프로젝트도 진행하며 다른 깨달음을 줄 수 있는 프로젝트도 병행하여 경험을 쌓아가면서 점점 이 게임을 개발 완료 단계로 만들어 주고 싶습니다. 그리고 아쉽게도 매우 극 초반에 기획한 게임이라 기획을 진행하면서 앱 시장 조사는 하지 못했지만 이 게임은 저의 첫 번째로 시장에 내놓게 되는 것에 의미가 있는 프로그램이 될 것이라 기대됩니다.

2-3 PROJECT 3 BINARY

NUMBER QUIZ

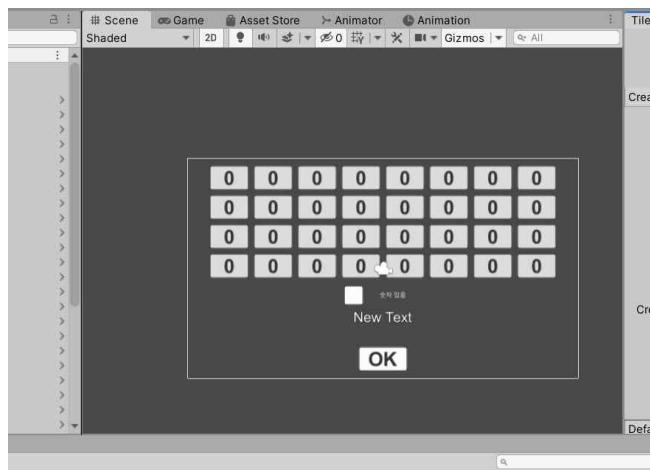
(1) 이 게임을 만들게 된 계기

2진수 변환법을 찾아보던 중 2진수 숫자들을 이용해 퀴즈를 만드는 글을 보게 되었습니다. 하지만 그 글은 8의 배수 숫자들 4개중에 하나를 골라서 맞추는 문제였습니다. 하지만 저는 더 범위를 넓혀서 2진수의 첫 번째 자리의 숫자가 1인 숫자들 중 하나를 골랐을 시 그 숫자를 맞추는 방법을 알아내 게임으로 만들어 보면 좋을 것 같다고 생각했습니다. 그리고 원래는 하나의 스크립트 안에 gameobject나 다른 스크립트를 모두 넣고 데이터마저도 하나하나 넣었던 것과는 다르게 해당하는 object가 나타내는 데이터를 따로 저장해주는 데이터파일을 따로 만드는 것이 좋다고 생각되어 데이터파일을 따로 만든 최초의 프로젝트입니다. 또한 전 프로젝트에서 object 관련 오류가 생겨 쓰지 못했던 배열을 적극 활용하기로 했습니다.

(2) 기본적인 게임 설명

이진수의 원리를 이용해 초반에 주어지는 카드 중에 유저가 생각한 카드를 맞추는 게임입니다.

(3) 기본 화면



0으로 입력되어있는 각 칸이 카드입니다. 그리고 첫 번째 index에서는 1부터 64까지의 소수 숫자 중에 하나를 골라 기억하라고 합니다. 유저가 기억하면 ok버튼을 눌러 다음 index로 넘어가게 됩니다. 두 번째 index부터 여섯 번째 index까지는 주어진 index안에 있는 카드 중에 자신이 기억하고 있는 카드가 존재하는지 판단하여 만약 존재한다면 사진에 보이는 toggle UI로 모두 체크하여 넘어가게 되며 끝까지 입력을 받으면 마지막에 더해진 sum값이 출력되어 “당신이 선택한 숫자는 sum입니다”라고 알려주게 됩니다.

(4) 베이스 코드

1번 이미지

```

33     /// <summary>
34     /// 메세지 텍스트
35     /// </summary>
36     string[] m_messageStr =
37     {
38         "숫자 하나를 선택한 후 기억해 주세요.",
39         "이 안에 선택하신 숫자가 있습니까?",
40         "당신이 선택한 숫자는[0:D]입니다."
41     };
42
43     /// <summary>
44     /// 인덱스
45     /// </summary>
46     int m_index = 0;
47
48     /// <summary>
49     /// 합
50     /// </summary>
51     int m_sum = 0;

```

2번 이미지

```

58
59     /// <summary>
60     /// 숫자 보기
61     /// </summary>
62     /// 참조 2개
63     void ViewNumber ()
64     {
65         m_messageText.text = GetMessage();
66
67         m_toggle.isOn = false;
68
69         if (m_index >= m_data.Length)
70         {
71             return;
72         }
73
74         for (int i = 0; i < m_numberBase.Length; i++)
75         {
76             m_numberBase[i].Setting(m_data[m_index], m_number[i]);
77         }

```

3번 이미지

```

79     /// <summary>
80     /// 메세지 생성
81     /// </summary>
82     /// <returns>메세지</returns>
83     /// 참조 1개
84     string GetMessage ()
85     {
86         if(m_index > 0 && m_index < m_data.Length)
87         {
88             if (m_toggle.isOn)
89             {
90                 m_sum += m_data[m_index - 1].m_number[0];
91                 m_toggleObj.SetActive(true);
92                 return m_messageStr[1];
93             }
94             else if (m_index == 0)
95             {
96                 m_sum = 1;
97                 m_toggleObj.SetActive(false);
98                 return m_messageStr[0];
99             }
100        }

```

1번 이미지

해당하는 상황의 텍스트를 뿐여주는 함수를 사용하였고 index는 해당하는 데이터 덩어리의 값 sum은 최종적으로 보여주게 되는 처음에 선택한 카드 번호가 무엇인지 알려주는 값입니다.

2번 이미지

for문을 통해 데이터파일에서 데이터를 불러와 각 카드에 뿐여주는 문장입니다. 만약 인덱스가 데이터 파일의 개수보다 커진다면 다시 되돌아갑니다.

3번 이미지

세 번째 이미지는 index의 값에 따라 사용자가 toggle의 체크여부를 검사해서 sum값에 알맞은 수를 더해주게 됩니다.

퀴즈 원리 설명

이 퀴즈는 2진수의 원리를 이용해서 만든 퀴즈입니다.

게임 내 데이터파일을 참고해서 설명하겠습니다.

NumberData0	NumberData1	NumberData2	NumberData3	NumberData4	NumberData5
Script	NumberData	Script	NumberData	Script	NumberData
▼ Number					
Size	32	Size	32	Size	32
Element 0	1	Element 0	4	Element 0	12
Element 1	3	Element 1	5	Element 1	4
Element 2	5	Element 2	6	Element 2	5
Element 3	7	Element 3	7	Element 3	7
Element 4	9	Element 4	10	Element 4	13
Element 5	11	Element 5	11	Element 5	13
Element 6	13	Element 6	14	Element 6	14
Element 7	15	Element 7	15	Element 7	15
Element 8	17	Element 8	20	Element 8	24
Element 9	19	Element 9	21	Element 9	20
Element 10	21	Element 10	22	Element 10	26
Element 11	23	Element 11	23	Element 11	23
Element 12	25	Element 12	28	Element 12	28
Element 13	27	Element 13	27	Element 13	28
Element 14	29	Element 14	30	Element 14	32
Element 15	31	Element 15	31	Element 15	31
Element 16	33	Element 16	36	Element 16	33
Element 17	35	Element 17	37	Element 17	41
Element 18	37	Element 18	38	Element 18	48
Element 19	39	Element 19	39	Element 19	42
Element 20	41	Element 20	42	Element 20	44
Element 21	43	Element 21	45	Element 21	46
Element 22	45	Element 22	46	Element 22	47
Element 23	47	Element 23	47	Element 23	47
Element 24	49	Element 24	50	Element 24	52
Element 25	51	Element 25	53	Element 25	57

2진수로 10진수를 구하는 예는 이렇습니다.

만약 101101이라는 2진수가 있다면

첫 번째 자리 2^0

두 번째 자리 0

세 번째 자리 2^2

네 번째 자리 2^3

다섯 번째 자리 0

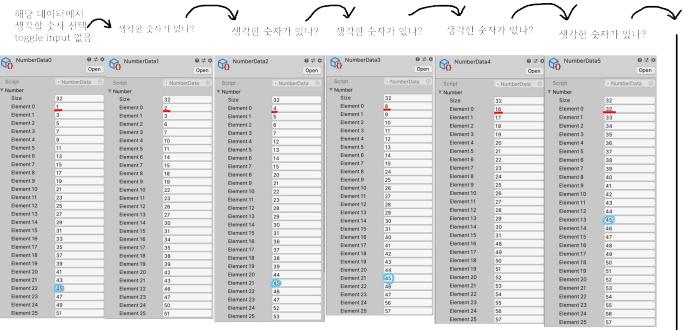
여섯 번째 자리 2^5

이 2의 제곱수들을 모두 더하여 10진수 45가 나오는 것입니다.

그리고 index의 카드들은 각 해당하는 자리수의 숫자들만 모아놓아져 있기 때문에 toggle에 체크를 했다면 2진수에 1이 표시되는 것처럼 각 체크마다 자리수에 알맞은 수를 sum값에 더해주는 것입니다.

그럼으로 표현하면 이렇습니다.

만약 생각한 숫자가 45일 경우



→ data0의 첫번째 자리 숫자인 1과 나머지 체크가 된 data들의 첫번째 값을 sum값에 추가

(5) 문제점

문제1. 2진수의 이해

본래 정보처리기능사를 공부하며 2진수를 변환하고 활용하는 방법을 알고 있었음에도 불구하고 분명히 원리를 잘 파악한 것 같으면서도 다시 보면 틀린 것 같고 2진수의 저주에 걸린 듯한 느낌이었습니다. 이해하는 단계에서 시간을 상당수 소모한 것 같습니다. 또한 프로그램으로서 옮기는 과정에서도 혼란이 있어서 다시 시작점으로 되돌아오게 되는 경험을 많이 한 것 같습니다. 하지만 반복적으로 공부하며 익혔습니다.

문제2. 데이터 선언 방법

데이터를 선언하기 위한 방법을 찾는데 또한 많은 시간이 소모되었습니다. 원래는 유니티 안에서만 계속 방법을 찾고 있었는데 스크립트로 데이터를 생성할 수 있는 메뉴를 따로 만들어줘야 한다고 해서 이런 방법도 있다는 것을 알아냈고 외국인 분들이 작성한 자료를 통해 벤치마킹에 성공했습니다. 앞으로도 유용하게 사용할 방법을 알아냈다는 점에 기분이 좋았습니다.

(6) 구현 화면

생각한 카드가 45인 경우를 가정해 실행합니다.

각 프래임마다 ‘OK’버튼을 누릅니다.

1	3	5	7	9	11	13	15
17	19	21	23	25	27	29	31
33	35	37	39	41	43	45	47
49	51	53	55	57	59	61	63

숫자 하나를 선택한 후 기억해 주세요.

OK

2	3	6	7	10	11	14	15
18	19	22	23	26	27	30	31
34	35	38	39	42	43	46	47
50	51	54	55	58	59	62	63

숫자 있음
이 안에 선택하신 숫자가 있습니까?

OK

4	5	6	7	12	13	14	15
20	21	22	23	28	29	30	31
36	37	38	39	44	45	46	47
52	53	54	55	60	61	62	63

숫자 있음
이 안에 선택하신 숫자가 있습니까?

OK

8	9	10	11	12	13	14	15
24	25	26	27	28	29	30	31
40	41	42	43	44	45	46	47
56	57	58	59	60	61	62	63

숫자 있음
이 안에 선택하신 숫자가 있습니까?

OK

16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

숫자 있음
이 안에 선택하신 숫자가 있습니까?

OK

32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

숫자 있음
이 안에 선택하신 숫자가 있습니까?

OK

32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

당신이 선택한 숫자는 45입니다.

OK

(7) 이 프로젝트를 진행하면서 느끼고 배운 점

전에 공부할 때는 시험 때만 기억할 수 있을 정도로만 공부해서 이 프로젝트를 하고 있을 때 즈음에는 많이 잊어먹은 단계였기 때문에 프로그램상으로 구현하기 위해 2진법 공부를 반복적으로 많이 했는데 이제 더 이상 머리에서 잊혀지지 않을 정도로 충분하게 공부한 것 같습니다. 또한 데이터파일을 따로 나누어서 활용하는 방법도 알아냈는데 이 방법은 앞으로 코딩을 진행하는데 일을 더욱 간단하게 처리할 편리한 방법 같았습니다. 그리고 코딩을 진행하고 다른 친구들에게 피드백을 받아가면서 변수선언 스타일을 바꾸었는데 이 방법이 전 세계의 프로그래머들에게 통용되는 방식인 것을 알고 앞으로의 게임 제작을 위한 다른 사람들 간의 커뮤니케이션에 유용하게 사용될 것 같아 저도 이 방식을 익히는 것이 좋을 것 같다고 판단되어 사용하면서 게임을 만들어 보았습니다. 이미 알고 있던 친구의 도움이 크게 영향을 끼쳐 고맙다고 생각했습니다. 그리고 전 프로젝트에서도 일부 변수를 바꾸어 보면서 변수선언을 익히게 되는 계기가 된 프로젝트였습니다.

2-4 PROJECT 4

CALCULATOR

(1) 이 게임을 만들게 된 계기

본 문서에 존재하는 두 번째 프로젝트를 진행하면서 string 문자열을 사용하여 매우 길게 짜야 할 코드를 획기적으로 줄인 경험이 있었습니다. 그 때문에 string 문자열을 앞으로의 활동에서 잘 활용하기 위해 간단한 프로젝트를 통해 한번 알아보면서 활용해 볼 수 있는 프로젝트를 한번 만들어보기 원했습니다. 그때 눈에 들어온 것이 백팩을 구매하기 위해 용량을 구할 때 사용한 휴대폰과 컴퓨터에 기본적으로 탑재된 계산기였습니다. 그 계산기를 보고 문득 이런 생각이 들었습니다. 왜 계산기는 프로그램상의 정해진 버튼을 눌러서 입력하는 방식밖에 없는지 그리고 string 문자열을 사용해서 사용자가 키보드로 어떤 방식으로 식을 입력해도 답을 산출해 낼 수 있을지 궁금했습니다. 이 프로젝트는 또한 앞으로 코드를 단순하게 만드는 곳에 유용하게 쓰일 열거형 enum을 적용해 본 프로젝트를 적용해 보기로 했습니다.

(2) 기본적인 게임 설명

보통 계산기는 직접 버튼을 누르는 방식이 대부분이지만 이 계산기는 유저가 키보드로 직접 쓴 최대 이항식의 식을 인식하여 계산하는 계산기입니다.

(3) 기본 화면



두 개의 항을 가진 식만 계산 가능합니다. 기호는 +, -, *, x, X, /, %를 사용 가능합니다. “Enter text”라고 써져있는 input값을 받는 텍스트 상자 안에 두 개의 항을 가진 식을 입력하고 계산을 누르면 “New Text”라고 써져있는 결과값 텍스트 상자에 결과가 산출되며 다시 한번 계산을 누르면 결과값이 input창으로 이동하면서 연속적인 계산이 가능합니다.

(4) 베이스 코드

1번 이미지

```

56     public void ButtonClick()
57     {
58         if (_n_canc == false)
59         {
60             if (_n_input.text == string.Empty)
61             {
62                 return;
63             }
64             if (_n_canc == false)
65             {
66                 if (_n_input.text.IndexOf('+', 1) > 0)
67                 {
68                     ChangeStr();
69                     _n_canc = true;
70                     Cstart();
71                 }
72                 else if (_n_input.text.IndexOf('/', 1) > 0)
73                 {
74                     ChangeStr();
75                     _n_canc = true;
76                     Cstart();
77                 }
78                 else if (_n_input.text.IndexOf('*', 1) > 0)
79                 {
80                     ChangeStr();
81                     _n_canc = true;
82                     Cstart();
83                 }
84                 else if (_n_input.text.IndexOf('-', 1) > 0)
85                 {
86                     ChangeStr();
87                     _n_canc = true;
88                     Cstart();
89                 }
90                 else if (_n_input.text.Replace("-", "+") > 0)
91                 {
92                     ChangeStr();
93                     _n_canc = true;
94                     Cstart();
95                 }
96                 else if (_n_input.text.IndexOf('%', 1) > 0)
97                 {
98                     ChangeStr();
99                     _n_canc = true;
100                    Cstart();
101                }
102            }
103            else if (_n_input.text.IndexOf('X', 1) > 0)
104            {
105                ChangeStr();
106                _n_canc = true;
107                Cstart();
108            }
109            else if (_n_input.text.IndexOf('x', 1) > 0)
110            {
111                ChangeStr();
112                _n_canc = true;
113                Cstart();
114            }
115        }
    
```

2번 이미지

```

124     //<summary>
125     /// <summary> argsStr>계산 문자열</para>
126     청조 2개
127     void SearchType (string argsStr)
128     {
129         string[] _in = string.Empty, string.Empty;
130         if (argsStr.IndexOf('+', 1) > 0)
131         {
132             _n_type = TYPE.Multiply;
133             _in = argsStr.Split('+');
134         }
135         else if (argsStr.IndexOf('/', 1) > 0)
136         {
137             _n_type = TYPE.Division;
138             _in = argsStr.Split('/');
139         }
140         else if (argsStr.IndexOf('*', 1) > 0)
141         {
142             _n_type = TYPE.Plus;
143             _in = argsStr.Split('*');
144         }
145         else if (argsStr.IndexOf('X', 1) > 0)
146         {
147             _n_type = TYPE.Multiply;
148             _in = argsStr.Split('X');
149         }
150         else if (argsStr.IndexOf('x', 1) > 0)
151         {
152             _n_type = TYPE.Multiply;
153             _in = argsStr.Split('x');
154         }
155         else if (argsStr.IndexOf('-', 1) > 0)
156         {
157             _n_type = TYPE.Division;
158             _in = argsStr.Split('-');
159         }
160         else if (argsStr.IndexOf('%', 1) > 0)
161         {
162             _n_type = TYPE.Minus;
163             argsStr = argsStr.Replace("-", "+");
164             if (argsStr.IndexOf('+', 1) > 0)
165             {
166                 SearchType(argsStr);
167                 return;
168             }
169             else
170             {
171                 _in = argsStr.Split('+');
172             }
173         }
174         else
175         {
176             _n_type = TYPE.None;
177         }
    
```

3번 이미지

```

175     switch (_n_type)
176     {
177         case TYPE.Plus:
178             _n_cin[0] = float.Parse(_in[0]);
179             _n_cin[1] = float.Parse(_in[1]);
180             GetPlus(_n_cin[0], _n_cin[1]);
181             break;
182         case TYPE.Minus:
183             _n_cin[0] = float.Parse(_in[0]);
184             _n_cin[1] = float.Parse(_in[1]);
185             Getminus(_n_cin[0], _n_cin[1]);
186             break;
187         case TYPE.Multiply:
188             _n_cin[0] = float.Parse(_in[0]);
189             _n_cin[1] = float.Parse(_in[1]);
190             GetMultipli(_n_cin[0], _n_cin[1]);
191             break;
192         case TYPE.Division:
193             _n_cin[0] = float.Parse(_in[0]);
194             _n_cin[1] = float.Parse(_in[1]);
195             GetDivision(_n_cin[0], _n_cin[1]);
196             break;
197         }
198     }
199     청조 1개
200     void GetPlus (float argA, float argB)
201     {
202         _n_text.text = (argA + argB).ToString();
203     }
204     청조 1개
205     void Getminus (float argA, float argB)
206     {
207         _n_text.text = (argA - argB).ToString();
208     }
209     청조 1개
210     void GetMultiply (float argA, float argB)
211     {
212         _n_text.text = (argA * argB).ToString();
213     }
214     청조 1개
215     void GetDivision (float argA, float argB)
216     {
217         _n_text.text = (argA / argB).ToString();
218     }
    
```

4번 이미지

```

219     //<summary>
220     /// 계산 시작
221     /// <summary> 청조 2개
222     청조 2개
223     void Cstart()
224     {
225         string _inStr = _n_input.text.Replace(" ", "");
226         Debug.Log(_inStr);
227         SearchType(_inStr);
228     }
229     //<summary>
230     /// 문장의 서로 빼기
231     //<summary>
232     청조 0개
233     void DstartCstr()
234     {
235         if (_n_text.text != string.Empty)
236         {
237             Debug.Log("입력값 검색기 버튼");
238             _n_input.text = _n_text.text;
239             _n_text.text = string.Empty;
240             return;
241         }
    
```

1번 이미지

if문을 통해 사용자가 입력한 식의 기호를 찾아 만약 존재한다면 시작할 때 해야할 일을 해주는 CStart 변수를 호출하고 계산을 계속 진행합니다. 없다면 ChangeStr함수를 호출합니다.

2번 이미지

if문을 통해 사용자가 입력한 식의 기호를 찾아 어떤 연산을 할지 최초로 결정합니다.

3번 이미지

switch case문에 따라 string문자열을 float변수로 바꾸어 직접 연산을 하는 함수로 전해줍니다.
연산하는 함수에서 연산한 뒤 다시 string 문자열로 바꿔 텍스트 상자에 넣어 줍니다.

4번 이미지

Cstart변수는 입력을 받은 문자열에 공백을 제거해
변수에 저장하고 연산을 진행하기위해
SearchType함수를 호출합니다.
ChangeStr함수는 입력값의 수들과 결과값의 수를
바꿔줍니다.

(5) 문제점

문제1. enum의 이해

enum은 함수도 아니고 변수도 아닌 것 같은 전에
겪어보지 못했던 방식인 것 같습니다. 때문에 이를
이해하고 적용하는 과정에서 힘을 조금 들였습니다.
enum은 값을 대입 불가능한 변수를 선언하는
느낌이라는 방식으로 이해가 되었습니다.

문제2. 함수의 복잡한 사용

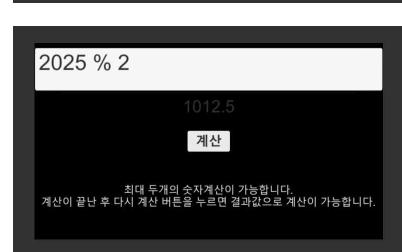
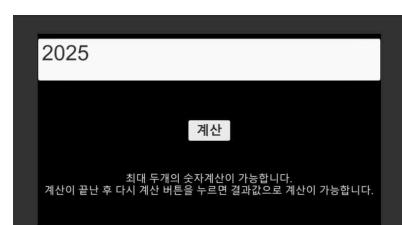
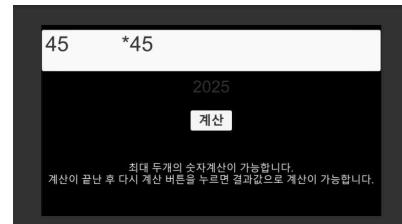
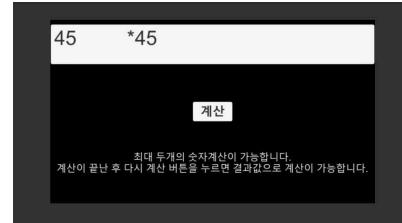
다른 프로젝트보다 함수 내에서 변수를 선언하는 방식을
통해 복잡하게 구성해 보았는데 가끔마다 보면 나중에
사용하는 함수 내의 변수가 내가 불러오고 싶은 값이
맞는지 혼란이 오기도 했었습니다. 베이스 코드의 경우로
예를 들면 입력값이 argStr -> in -> cin 변수로 나중에
실행되는 코드로 가며 바뀌게 되는데 다른일을 하다가
다시 코딩을 하게 되면 헷갈리는 경우가 있었습니다.
하지만 많은 경험을 통해 이를 극복해야 할 것 같다고
생각했습니다.

문제3. 완벽의 모순

전의 프로젝트에서 생각이 나지 않는 부분은 매우 많은
시간을 들여 생각해야 했었는데 이 프로젝트에서는 오류
해결이 너무 수월했고 더 이상의 오류는 발견되지
않았습니다. 한편으로는 성장한 것 같아 기분이 좋기는
했지만 프로그래머들 사이에서 오류의 부재는 심각한
일이라는 것을 알고 있었기 때문에 검토를 여러번
했더니 생각하지 못했던 상황을 통해 오류는 생기지
않더라도 처리가 불가능한 경우를 생각해 내게
되었습니다. 만약에 유저가 '*'가 아니라 'x'를 사용하는
다른 부호를 사용하는 경우들 등의 문제가 보였습니다.
프로그래밍은 완벽한 경우가 없다는 점이 확실히 와닿은
것 같습니다.

(6) 구현 화면

각 프레임마다 '계산'버튼을 누릅니다.
계산을 원하는 이항식을 기호와 함께 입력합니다.



(7) 프로젝트를 구성하며 느끼고 배운 점

문자열을 자동으로 해석하여 계산해주는 알고리즘을
만들어 보았더니 왜 대부분 계산기가 버튼을 누르는
식으로 직접 숫자와 기호를 입력하는 방식으로 돼
있는지 알 것 같습니다. 특히 제한이 없는 다항식을
처리해 주려고 알고리즘을 생각했는데 굉장히
복잡해졌습니다. 특히 곱하기와 나누기 우선 처리와
기호를 기준으로 나눔으로서의 중복계산 등이 문제가
되었습니다. 시간이 오라 걸릴 것 같아 지금은 재현하지
않을 예정이지만 추후 시간이 된다면 한번 만들어 보고
싶습니다. 또한 프로젝트는 문자열의 사용으로 코드를
간단하게 만드는 것이 주 목적이었는데 완벽하게는
아니지만 앞으로 일반적인 상황에서 쓰일 방식은 대부분
익혔으므로 목적을 달성했다고 볼 수 있었습니다. 그리고
함수의 복잡한 사용을 통해서 같이 할 때뿐만 아니라
혼자 프로젝트를 진행할 때도 주석이 중요한 역할을
한다는 것을 깨달은 것 같습니다. 이런 과정에서
성취감을 느꼈고 추가로 Enum 열거형 방식도 학습
가능해서 다양한 경험을 해본 좋은 시간이었던 것
같습니다.

3-1 그래픽 작업을 시작 한

계기

저는 초등학생 때 방과 후로 미술을 배우면서부터 자기 생각을 그림으로 표현하는 것을 좋아했습니다. 그리고 예술성에 상상력까지 발휘하여 심심할 때마다 그리면서 만화책을 하나 만들어서 반 친구들에게 보여주었고 친구들의 반응은 굉장히 좋았습니다. 그때부터 예술에 관심을 두기 시작한 것 같습니다. 그리고 학년이 올라가고 친구들은 저와 다른 친구들의 미술 작품은 놀랄지언정 저의 미술 작품은 놀리지 못했습니다. 그렇게 미술 과목은 좋아하는 과목 중 하나로 여겨져 왔습니다. 그러던 중 저에게 컴퓨터라는 기계가 등장하면서 미술은 저에게 다른 방식으로 다가왔습니다. 물론 처음에는 게임기로 쓰였지만 배우면서 컴퓨터의 활용도가 얼마나 넓은지 깨닫게 되었습니다. 그리고 결정적인 계기로 동영상 사이트에서 포토샵이라는 프로그램으로 다른 곳에서 붙여오고 수정하는 방식을 이용해서 하나의 비행기가 추락하는 한 장면의 작품을 만드는 것을 보고 멋지다고 생각할 때부터였습니다. 그때부터 저도 그래픽 창작이라는 것을 시작하게 되었습니다. 하지만 막상 시작하려니 무엇을 만들지 고민되었고 인터넷에 여러 가지 공모전 사이트를 찾아다녔습니다. 하나는 국가에서 주최하는 공모전과 기업에서 주최하는 공모전이 주를 이뤘고 하나는 의뢰자가 디자이너들에게 자신이 원하는 디자인을 신청하고 그중 마음에 드는 디자인을 골라 의뢰비를 주는 시스템이었습니다. 저가 이 두 사이트를 선택한 이유는 포스터와 로고 디자인이라는 비교적 간단한 작업이라는 것이기 때문이었습니다. 또한 게임프로그래머로서 그리고 4차산업을 이끌어갈 사람으로서 간단한 그래픽 작업쯤은 기본적으로 할 줄 알아야겠다는 그런 생각이 들었습니다.

3-2 포스터

에너지 절약 포스터

이제 그만

빼도 되지 않겠어요?



무의미한 에너지 사용은
지구를 빨아 먹는 행동입니다.
지구에게 회복할 시간을 주시는건 어떤가요?

전기 에너지의 낭비는 지구를 해치는 것이라는 것을 표현하고 싶었습니다. 나 한 사람이라는 안일한 생각 때문에 쓰지도 않는 에너지를 낭비하는 것은 개인적으로 볼때나 사회적으로 볼때 바르지 못한 행동입니다.

도박 예방 포스터

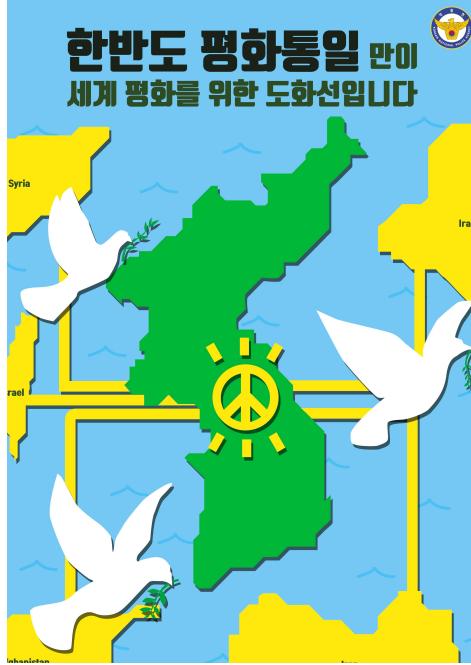


포커에서 카드를 뒤집는 행위를 이용해 도박을 하는 자신의 인생도 쉽게 뒤집을 수 있겠냐고 묻는 포스터입니다. 카드를 뒤집는 말을 더욱 잘 표현했으면 더 좋은 작품이 될 수 있을 것 같은데, 아쉽습니다.

평화통일 포스터

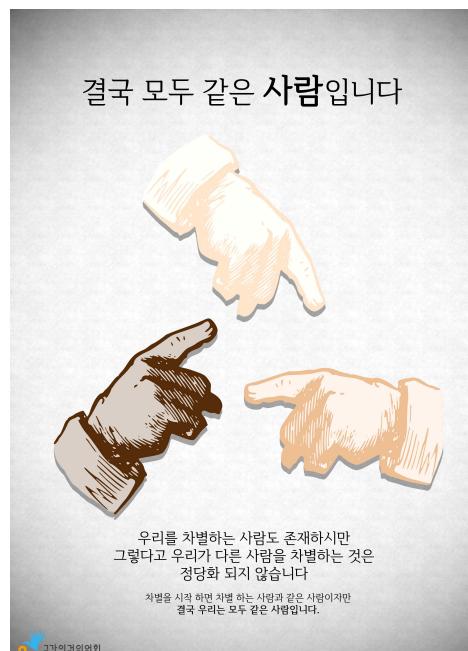
3-3 로고

이름을 제외한 모든 창작물은 창작자에게 저작권이 있습니다.



살짝 멋진 감이 있지만 다른 작품들과는 다르게 개성 있는 그림에 힘을 쏟았습니다. 하지만 공모전이 갑자기 사라져버려 무용지물이 되어버린 포스터라 아깝다고 생각합니다.

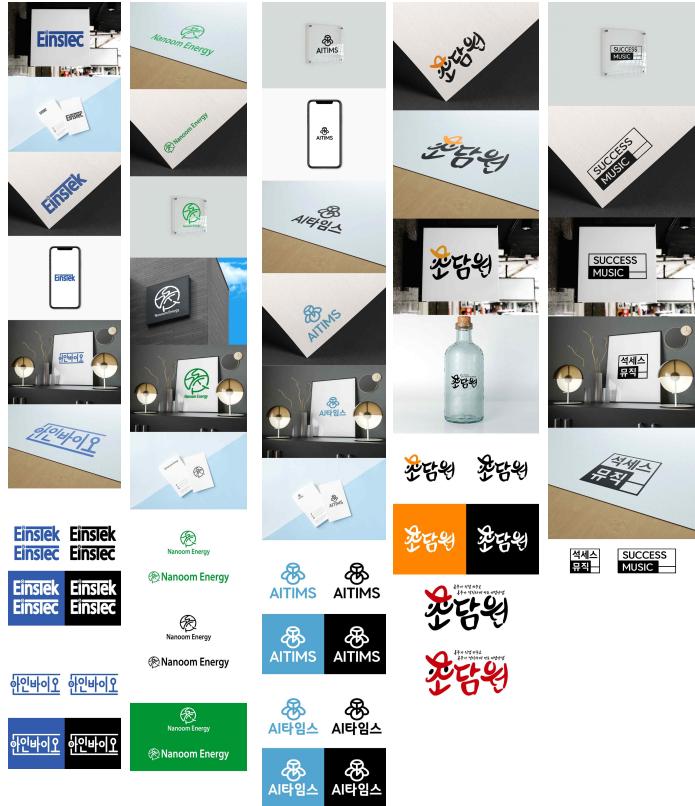
인종차별 포스터



인종 차별없이 우리 모두는 같은 ‘사람’이기 때문에 서로 차별해서는 안되고 우리나라의 경우도 인종차별 사례가 충분히 존재하기 때문에 우리를 차별하는 사람과 결국 ‘같은 사람’ 일 수도 있으니 인종차별을 하지 말자는 포스터입니다.



왼쪽부터 순서대로 : 한의원, 고깃집, 옷가게, 식기회사, 의료기기 회사, 자동차 잠금 해제 앱, 샌드위치 가게, 아파트, 교회, 망고 가게, 코딩 단체



왼쪽부터 순서대로 : 학원, 에너지 사업체, AI연구소 식료품 업체, 피아노 학원

3-4 그래픽 작업을 하며

느낀 점

작업하면서 저는 만드는 스타일에 큰 변화가 생겼습니다. 그리고 스타일뿐만 아니라 작업 속도도 눈에 띄게 빨라졌고 포스터보다 간단했던 로고는 생각만 나면 한 시간 안에 금방 만드는 상황에까지 이르렀습니다. 또한 문제를 해결해 나갈 때 도움을 줄 여러 교훈도 깨닫게 되었습니다. 첫 번째로는 분석의 중요함입니다. 실패에 따른 분석은 저를 달라지게 만들었습니다. 그리고 초등학교 때 썼던 오답 노트도 마찬가지로 문제의 정답률을 달라지게 만들었죠. 저는 이미 실패에 따른 분석을 오래전부터 해오고 있었던 것입니다. 하지만 그래픽 작업을 진행하기 전까지는 그것이 얼마나 중요한지 모르고 있었습니다. 다른 사람이 상을 탄 경우를 통해 나의 작품에는 무슨 이유가 있어서 떨어졌는지 원인을 분석하고 제삼자의 입장에서 바라보니 나 자신이 잘 만들었다고 생각했던 작품에 허점이 수두룩하게 보이기 시작하는 것이었습니다. 매우 신기한 경험이었습니다. 분명 만들고 난 당시에는 너무 잘했다고 생각했는데 나중에 보니 가독성도 떨어지고 무슨 말을 하는 건지도 잘 모르겠다는 것이었습니다. 특히 '에너지 절약 포스터'와 '도박 예방 포스터'에서 확실히 느껴졌습니다. 그리고 최근 상을 받는 포스터 스타일로 마지막 포스터인 '인종차별 포스터'를 만들어 보았는데 아직 부족한 점이 보이지만 전보다는 확실히

나아졌음을 느꼈습니다. 두 번째는 자기가 기본적으로 잘하는 것이 중요할 수도 있지만 노력과 경험이 더욱 중요하다는 것입니다. 사실 처음 포스터와 로고를 만들 때는 “보기 나쁘지는 않은 것 같은데 하나쯤은 얻어걸리겠지”라는 생각으로 만들었습니다. 그렇지만 포스터도 그랬듯이 마찬가지로 로고 작업도 일부 자잘한 경우를 빼면 실패의 연속이었습니다. 그 때문에 작품을 분석하기로 했습니다. 로고 디자인 사이트를 통한 공모와 포스터 공모전이 조금 달랐던 점은 의뢰자분들에게 선택받으신 디자이너분들의 작품을 감상할 수 있다는 것입니다. 저의 작품에서의 문제점을 찾고자 다른 디자이너분들의 작품을 감상해 보았습니다. 선택받은 작품들은 대부분 “이런 생각을 어떻게 하는 거지?”라는 말이 절로 나오게 했습니다. 그리고 더욱 많이 찾아보면서 또 다른 공통점이 있었는데 선택은 대부분 전에 선택을 많이 받은 사람이 받는 편이었고 자신이 작업한 mokup(적용 예시 사진) 밑에 회사 느낌이 나는 로고도 하나씩 붙어있는 것이었습니다. 처음에 봤을 때 중소기업에 다니시는 분들이 용돈으로 벌고 있는 줄 알았지만 알고 보니 이 회사는 로고 전문 제작업체였습니다. 한편으로는 회사가 개인이 하는 일에 끼어드는 건 너무 반칙 아닌가 라고도 생각했고 한편으로는 최근 웹디자인부터 로고 디자인까지 조금만 만들어 보면 괜찮은 결과가 나오는 쉬운 디자인들은 쇠퇴기에 접어들었기 때문에 만약에 그냥 만들어 보려고 온 사람이 선택받으면 일자리를 뺏게 되는 것이 아닌가 하는 죄송한 감정도 들었습니다. 그런데 그리고 보면 이분들은 이런 간단하게 생각했던 분야에서 전문화된 것이었습니다. 더 찾아보니 로고를 어떻게 만들어야 하는지에 대한 책도 있었고 강의도 있었습니다. 이런 분야도 전문화가 되었다는 사실은 자기의 기본적인 실력만 믿고 선택을 받으리라 생각했던 나 자신에 큰 교훈을 주었습니다. 이렇게 그래픽 작업은 처음에는 상상하지도 못했던 교훈들을 주었습니다. 이 교훈들은 어떤 일을 하던 다시 한번 나 자신을 돌아보게 해줄 것 같습니다. 그리고 그래픽 디자인이 직업이 될지는 모르겠으나 할 수 있는 것과 할 수 없는 것은 천지 차이기 때문에 앞으로의 프로그래밍 능력에 영향을 끼칠 수밖에 없습니다. 그리고 더 나아가 이번에 얻은 교훈과 기술을 통해 모든 부서를 총괄할 CEO가 되는 꿈에도 한 발짝 나아가게 되는 좋은 경험이 된 것 같습니다.

4-1 게임 간단 기획서

* 세부 계획서가 따로 있지만 분량 제한 때문에 간단 계획서만 첨부했습니다.

달리자 마라톤 게임계획

계획 팀 : 마라토너



"무슨 일이 있어도 끝까지 달린다!"

달리자 마라톤

목차

게임개요	1
게임특징	2
게임진행	3

게임개요

게임 제목 : 달리자 마라톤

- 마라톤을 뛰는 게임이라는 것을 단번에 알 수 있다.
- 이를 이용해 마라톤에 관심있는 사람부터 마라톤에 관심 없는 사람까지 흥미를 느낄 수 있다.

기획의도

- 기존 속도감만 중요시 한 레이싱 게임들의 전형적인 방식이 조금 마음에 들지 않았다.
- 기존 조깅이나 마라톤에 관심이 있던 사람들이 재미있고 쉽게 마라톤을 접할 수 있게 하고 싶었다.

게임특징(흥미요소)

- 기존 게이밍과는 다르게 판경과 체력에 따라 전략적인 접근이 필요한 게이밍 게임이다.
- 멀티플레이와 싱글플레이어가 나누어져 있어 다양한 유형의 플레이어들을 끌어 들일 수 있다.
- 레이싱게임에 소特有的 요소와 다양한 스키in 등을 통하여 자신의 캐릭터를 키운다는 느낌을 받을 수 있으며 무과금 시에도 충분히 그 기분을 느낄 수 있다.
- 디모바일 게임과는 다르게 다양한 모드가 존재하고 그 모드들은 유저가 설정 가능하다.

플랫폼

- 모바일 안드로이드 및 iOS

타겟연령층

- 성인플레이어 (연령층 = 8 ~ 12)
- 멀티플레이어 (연령층 = 12 ~ 19)

- 3 -

게임특징

1. 싱글플레이 요소

이 게임은 멀티플레이어 뿐만 아니라 싱글플레이도 존재하고 싱글플레이 스토리가 상당히 흥미롭다.

2. 다양한 스키in

다양한 마라토너 캐릭터를 통해 재화를 통해 구매가 가능하고 마라토너의 옷도 스키in으로 판매를 하여 늘은 커스터마이징 자유도를 지니고 있다.

3. 다양한 환경요소

맵에 따라서 건조한 맵의 경우는 언덕 그리고 산지의 경우 들 등 다양한 장애물들이 존재하고 상황마다 비, 눈, 우박 등의 날씨가 존재하며 낮과 밤같은 판경요소들에 따라 캐릭터의 소켓 및 고유속성에 따라 지령에 대한 디버프와 버프가 존재하여 전술적으로 캐릭터를 선택하는 판단력 또한 필요해 흥미진진하다.

3. 능력치 시스템

게임 내 재화로 구매한 캐릭터마다 고유 능력과 레벨에 따른 소켓 시스템이 존재하여 게임 초반에는 멀티플레이어도 게임 맵을 느리게 하여 초보자들의 게임 유입을 도우면서 자기만의 캐릭터를 육성한다는 느낌을 받을 수 있는 능력치 시스템이 있다.

4. 미니게임, 복불복

체력을 채우는 각 식량 배급소마다 다양한 미니 게임과 목물목이 존재하여 유저들의 흥미를 유발하고 경쟁심에 더욱 물을 끓이게 되는 요소가 존재한다.

4. 다양한 모드

게임의 특징을 살린 다양한 모드가 존재하고 또한 그 모드들을 통해 유저가 마음대로 설정 할 수 있어 질리지 않는 게임을 경험할 수 있다.

- 4 -

- 2 -

게임진행

멀티플레이 게임 모드 중 대표적인 모드인 마라톤 모드



처음에 플레이어들에게 달릴 장소의 지명과.....
거리, 날씨 등의 정보를 제공해 준다.
정보를 제공....방문을 플레이어들은 준비시간에
자기의 상황과 맞는 캐릭터와 장비를 선택하여
게임 준비를 마쳐야 한다.



플레이어는 조이스틱을 이용해 마라토너의 x좌표를
이동시키거나 달리고 멈추게 할 수 있고 점프
버튼을 사용하여 장애물을 피하는 방법도 가능하다.
배급소에 방문해 독물독이나 미니게임을 수행하여
이기게 되면 식량 배급을 받아 체력을 채워야
한다. 또한, 각 플레이어들은 마라톤 시 다른
플레이어들을 장애물 설치, 디파우트 적용 등으로
방해하는 것이 가능하며 공격을 받은 플레이어는
공격에 따라 방어하는 방법이 존재한다.



체력이 모두 다해거나 중간에 나가 포기하게되면
마지막 결승점까지 다양한 경우의 수에 따라 1등
2등 3등까지 결정되고 게임 한 판은 모든 주자가
탈락하거나 들어와야 비로소 한 판이 된다.
순서대로 1등, 2등, 3등, 특정한 시간 내에 종과,
시간 무관하고 종과 순으로 게임내 재화를
지급받게 된다. 또한, 전경에 기록되게 되어
성취감을 느낄 수 있고 뒷수에 따라 추후 게임
정식 대회 출전권이 주어지게 된다.

4-2 게임 기획서의 제작 계기와 배우고 느낀 점

전에 해왔던 것과 같이 각종 공모전을 검색하고 있을 때 우연히 이러한 대회를 알게 되었고 국가가 개최하는 게임 제작, 기획 관련 대회도 다른 분야보다 많이 없는 편인 것을 알고 있었기 때문에 좋은 경험과 기회가 될 수 있으리라 생각하여 한 번 도전해 보기로 했습니다. 지금까지 게임을 만들어 오면서 간단한 기획과 사전조사는 많이 해봤지만 이렇게 세부적으로 계획하는 경우는 없었습니다. 하지만 이번 경험을 통해서 저 같은 프로그래머도 기획에 참여할 수 있는 4차 산업혁명에 걸맞는 프로그래머가 되고 싶었습니다. 예전의 프로그래머들과는 다르게 자신에게 주어진 도구를 통해 활용만 한다면 좋은 효율을 낼 수 있습니다. 그리고 그러한 정세에 따라 소규모로 개발을 진행하는 경우가 늘고 있습니다. 게임을 만드는 기업들은 게임을 개발하는 사람을 뽑을 때 프로그래머를 뽑는다고는 하지만 다른 작업도 어느 정도 이해할 수 있는 유동성 있는 사람을 뽑으려고 하고 있다는 것을 알고 있었습니다. 그리고 고용되는 처지에서 고용하는 처지로 시선을 바꾸어도 이런 능력을 갖추는 것은 직원을 뽑을 때나 관리할 때 모두 유용하게 쓰일 수 있을 것 같다는 생각도 들었습니다. 그 때문에 이번 기획해 본 경험이 앞으로

도움을 줄 수 있으리라 생각합니다.

간단 기획서와 세부 기획서를 모두 작성해 제출했는데 간단 계획서는 쉬운 편이었지만 세부 계획서에서 정체가 왔었습니다. 시작할 때 총 20면을 써야 한다는 생각에 굉장히 막막했습니다. 하지만 시간은 많았기 때문에 천천히 무엇을 써야 할지부터 생각해 보았습니다. 그래서 늘 정보를 찾던 방법으로 구글 검색을 해 보았는데 예상과는 달리 결과가 나오지 않는 것이었습니다. 기획서는 인터넷에 잘 올라오지 않는 것 같아 포기하고 그래픽 작업에서 활용해 보았던 전에 우승했던 분들의 목차를 벤치마킹하는 방법과 짧게 기획을 해봤던 경험을 참고하여 기획서에 쓸 것들을 정리해 나가기 시작했습니다. 처음에 목차를 작성함을 통해 무엇을 쓸지 정리를 잘해 두니 처음에 막막했던 감정이 무색하게 작성할 수월히 할 수 있었습니다. 하지만 문제가 하나 생겼습니다. 바로 분량 문제와 대회 목적에 대한 회의감이었습니다. 게임을 세부적으로 기획하다 보니 끝이 없을 것 같았습니다. 그래서 저는 얼마나 세부적인 내용까지 적을지 기준을 정했습니다. 내용에서 너무 세부적으로 들어갈 수 있는 내용이 세부적인 능력치라던가 확률 등의 수학적인 부분에서의 관계를 설명하는 부분과 세부적인 그래픽에 대한 부분이었습니다. 그리고 이 대회는 게임의 특징을 잘 드러내는 기획서를 원하는 것 같았기 때문에 게임의 특징을 나타내고 설명하는데 불필요한 부분을 과감히 생략했습니다. 물론 그렇지 못한 부분도 일부 존재하지만 분량을 맞출 수 있었습니다. 잘 되어 가고 있던 중 세부 기획서 작성은 마치기 바로 전 저는 심각한 문제를 깨달아 버렸습니다. 바로 양식이 따로 정해져 있던 것입니다. (대회에 제출한 기획서는 본 기획서와 양식이 다릅니다) 저번 년에도 자유롭게 작성했던 작품들의 목차를 벤치마킹하다 보니 공지사항에 있던 양식을 지키지 않고 작성해 버린 것입니다. 그리고 간단 기획서도 마찬가지였습니다. 저는 일단 동요하지 않게 평정심을 유지했습니다. 그리고 시간 내에 양식을 알아 치울 수 있을지 침착하게 생각해 보았습니다. 마감까지 하루 정도 남았으니 가능하다고 판단했습니다. 그리고 지금 와서 포기하기에는 너무 아까웠습니다. 그래서 시간 내에 완료하겠다는 신념으로 시간 안에 양식에 맞춰 대회에 제출하게 되었습니다.

기획서를 작성하는 중 내내 들었던 생각은 “내가 지금 하는 것이 알맞게 하는 것인가?”라는 걱정이었습니다. 왜냐하면 기획서를 처음 만들어 본 것이기 때문입니다. 하지만 그 걱정은 결과가 나왔을 때 눈 녹듯이 사라졌습니다. 결과를 통해 저는 나에 대한 신념을 느낄 수 있었습니다. 결국 답은 다른 사람들을 따라 하는 것이 아니라 자기가 보여주기 원하는 것을 표현하는 것에 대한 차이라고 결론 지을 수 있었습니다.