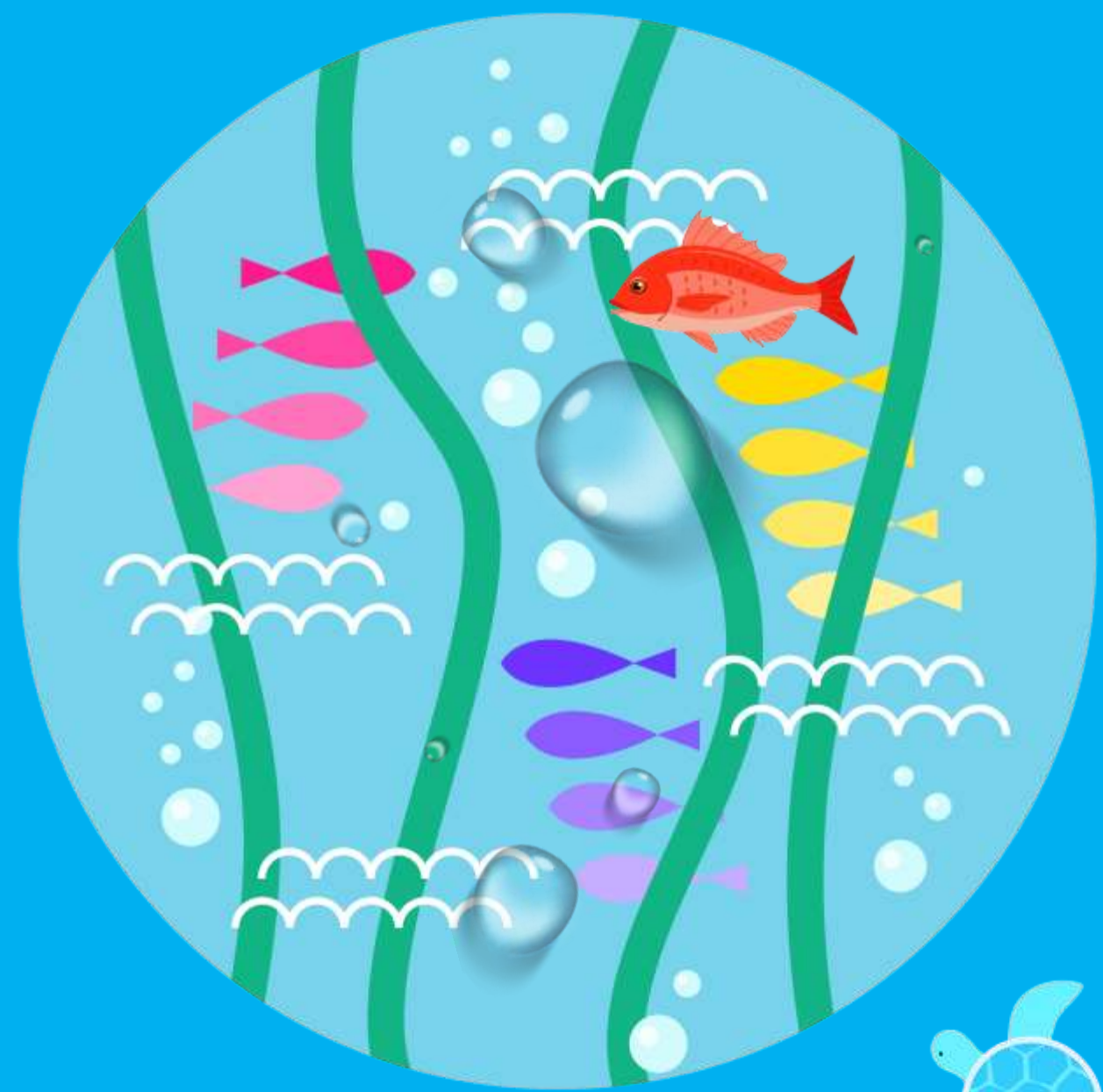


바다를 지켜라

- 해양 오염 개선을 위한 게임 개발 -

전 유 진



목차 INDEX

1. 기획 및 시나리오
2. 플로우 차트
3. 게임 소개 및 주요 기능
4. 소스 코드
5. 향후 개선 방향
6. 마케팅 활용 방안



1. 기획 및 시나리오

1. 게임 소개 및 주요 기능

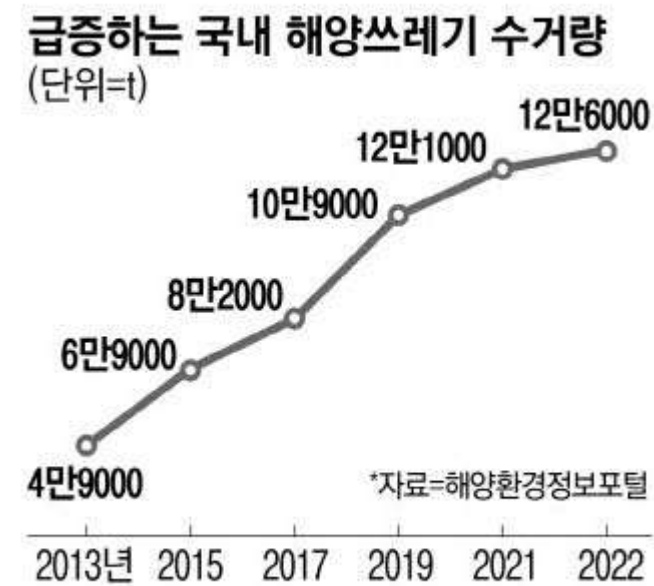
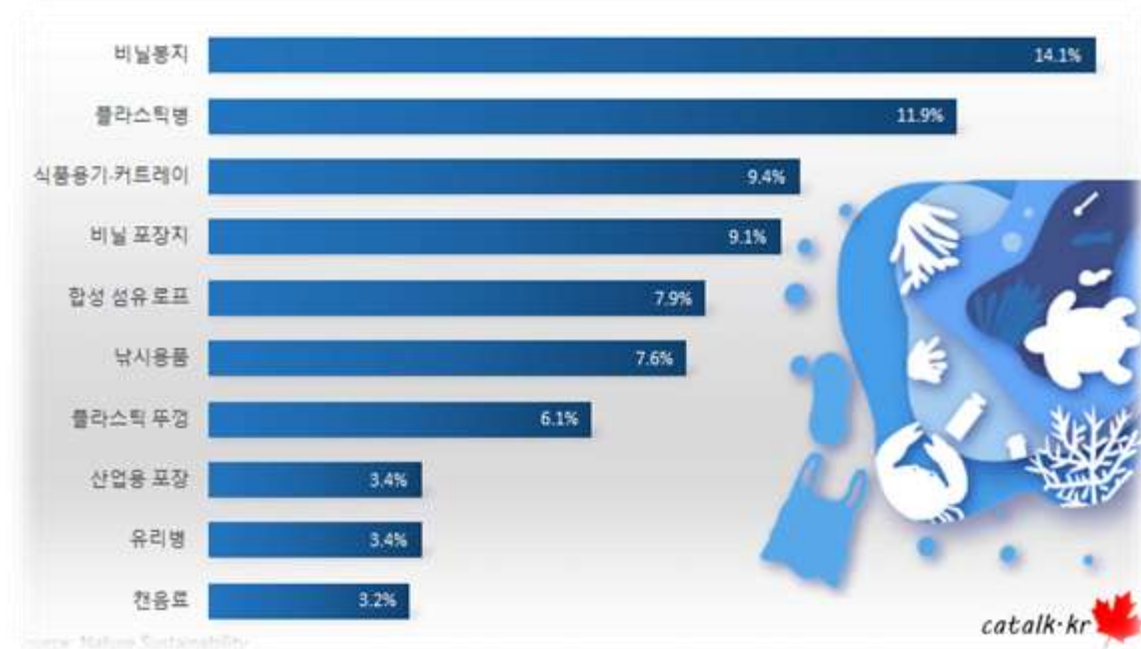
2-1) 목적 및 시나리오



어업을 하며 생계를 잇고 있는 김철수 씨.
해양 수질 악화로 어획량이 크게 감소해 피해를 입게 되는데...
김철수 씨의 **미션**은,
해양 쓰레기를 수거하여 깨끗한 수질 등급을 만드는 것!
바다를 지켜라!

1. 기획 및 시나리오

2-2) 실태 파악



- 10년 동안 해양 쓰레기 **지속적 증가**
- 지난해 최대치 **12만 6천** 톤 기록, 각종 문제점 야기
(선박 사고 유발, 어업 생산성 저감, 수거·처리 비용 발생 등)
- 해양 쓰레기로 인한 **산업적 피해**와 **처리 비용** 연간 3800억 원

1. 기획 및 시나리오

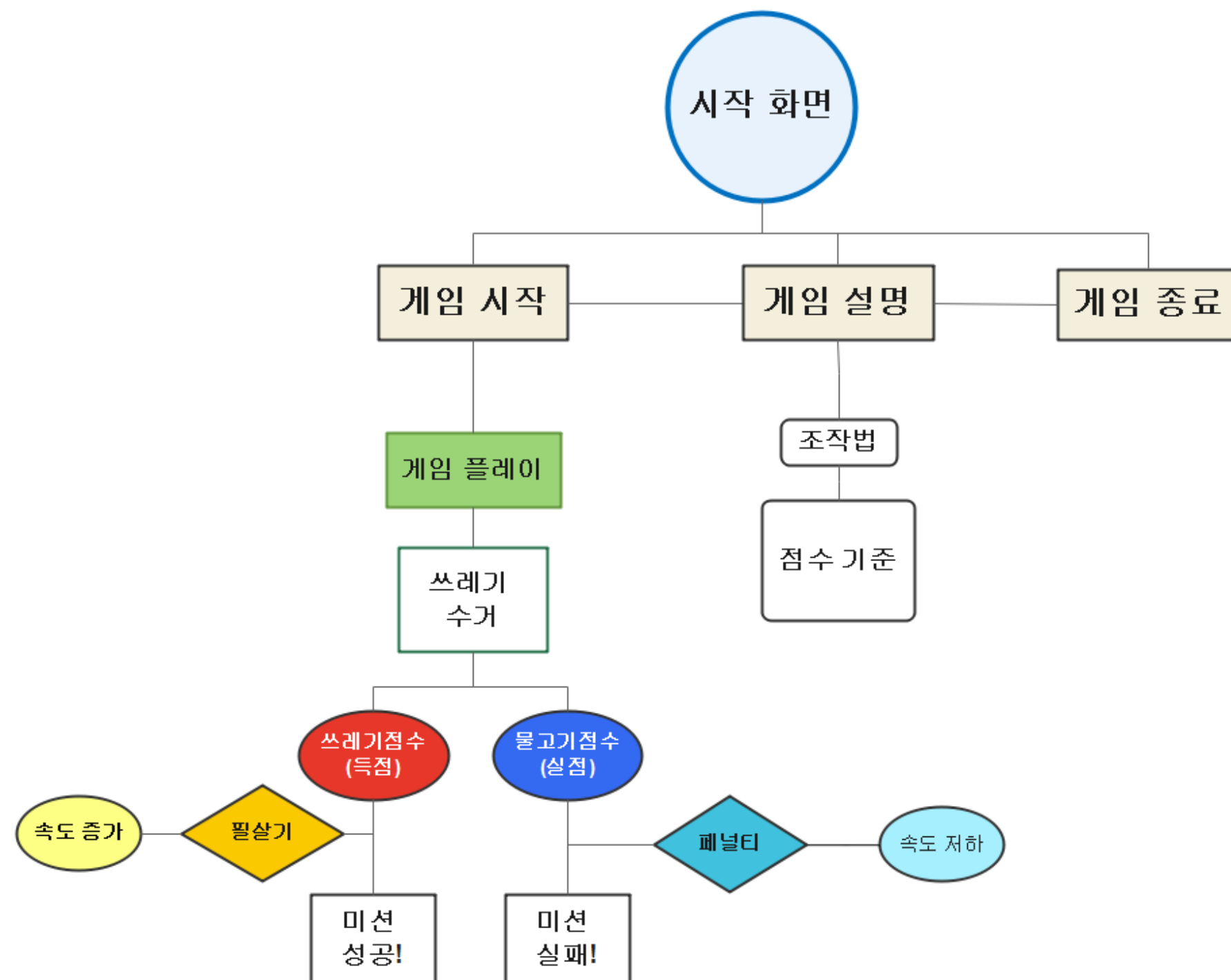
2-3) 서비스 기획



- 2015년 제주에서 시작된 '바다 환경 지킴이' 제도의 전국적 확대에 착안
- 해양 생태계 관심과 범 국민적인 환경 인식을 함양하기 위한 게임 콘텐츠 기획

2. 플로우 차트

2. 플로우 차트



3. 게임 소개 및 주요 기능

3. 게임 소개 및 주요 기능

3.1 오프닝 화면



1 게임 시작

2 게임 설명

● 조작법, 아이템, 필살기, 승부 방법

3 게임 종료

3. 게임 소개 및 주요 기능

3.2 게임 설명 화면

조작법 1



위, 아래로 이동

점수 기준 2

등급	수질평가지수 Water Quality Index	색깔
1 매우 좋음	0 - 23	Blue
2 좋음	24 - 33	Light Blue
3 보통	34 - 46	Yellow
4 나쁨	47 - 59	Orange
5 아주 나쁨	60 이상	Red

0점이 되면 성공!
100점에서 시작!
120점이 되면 실패!

쓰레기 점수 3



7종 각 - 10점

물고기 점수 4



8종 각 + 5점

필살기 5

60점 도달시
플레이어 속도 증가

페널티 6

115점 도달시
플레이어 속도 저하

- 1** 조작법
- 방향키 - 위, 아래로 이동

- 2** 점수 기준- 수질 평가 지수
- 5개 등급에 따라
파랑, 하늘, 노랑, 주황, 빨강 폰트
 - 0점이 되면 미션 성공!

- 3** 쓰레기 점수
- 7종, 각 10점 차감
 - 120점이 되면 미션 실패!

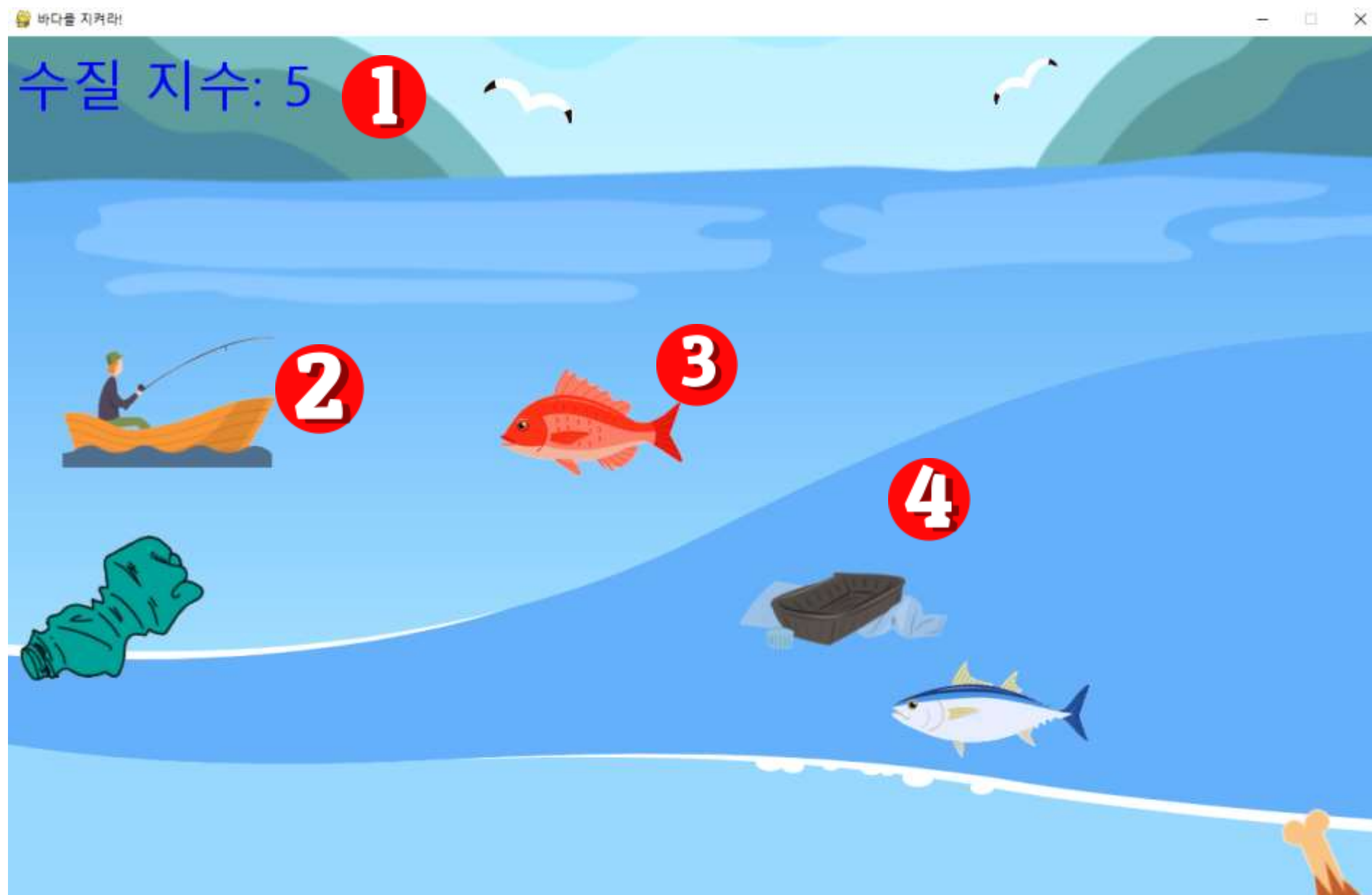
- 4** 물고기 점수
- 8종, 각 5점 증가


- 5** 필살기
- 60점 도달 시 플레이어 속도 증가

- 6** 페널티
- 115점 도달 시 플레이어 속도 저하

3. 게임 소개 및 주요 기능

3.3 게임 소개



- 1** 점수
 - 수질평가지수 WQI(Water Quality Index) 
- 2** 캐릭터 위치 / 이동
 - 상하 이동
- 3** 물고기 위치 / 이동
 - 오른쪽에서 등장, 좌측으로 전진 이동
- 4** 쓰레기 위치 / 이동
 - 오른쪽에서 등장, 좌측으로 전진 이동

3. 게임 소개 및 주요 기능

3.4 엔딩 화면

미션 성공 화면



미션 실패 화면



4. 소스코드

4. 소스코드

4.1 게임 소스코드 구조

음악 및 이미지 로드

- #01. 화면 크기 설정
- #02. 음악 사운드 로드
- #03. 메인 화면 이미지 로드
- #04. 게임 설명 이미지 로드
- #05. 메인 화면 버튼 이미지 로드
- #06. 화면 나가기 이미지 로드
- #07. 메인 게임 화면 로드
- #08. 플레이어 이미지 로드
- #09. 물고기 이미지 로드
- #10. 물고기 기본 위치 설정
- #11. 쓰레기 이미지 로드
- #12. 쓰레기 기본 위치 설정
- #13. 게임 클리어 화면 로드
- #14. 게임 오버 화면 로드

이미지 그리기

- #15. 게임 클리어 및 게임 오버 화면 등장 함수 정의
- #16. 메인 화면 버튼 기본 값 설정
- #17. 메인 화면 버튼 기본 값 설정
- #18. **Button** 클래스를 활용하여 버튼 객체 설정

기능 설정

- #19. 화면 나가기 버튼 객체 생성
- #20. 버튼 이미지 3개 그리는 함수
- #21. 메인 화면과 버튼을 그리는 함수
- #22. 설명 화면과 나가기 버튼을 그리는 함수
- #23. 메인 화면에서 버튼을 클릭했을 때 작동하는 함수
- #24. 화면 나가기 버튼 눌렀을 때 작동하는 함수

기타 변수 설정

- #25. **WQI**[수질오염지수]에 따른 색상 변환 함수

게임 플레이

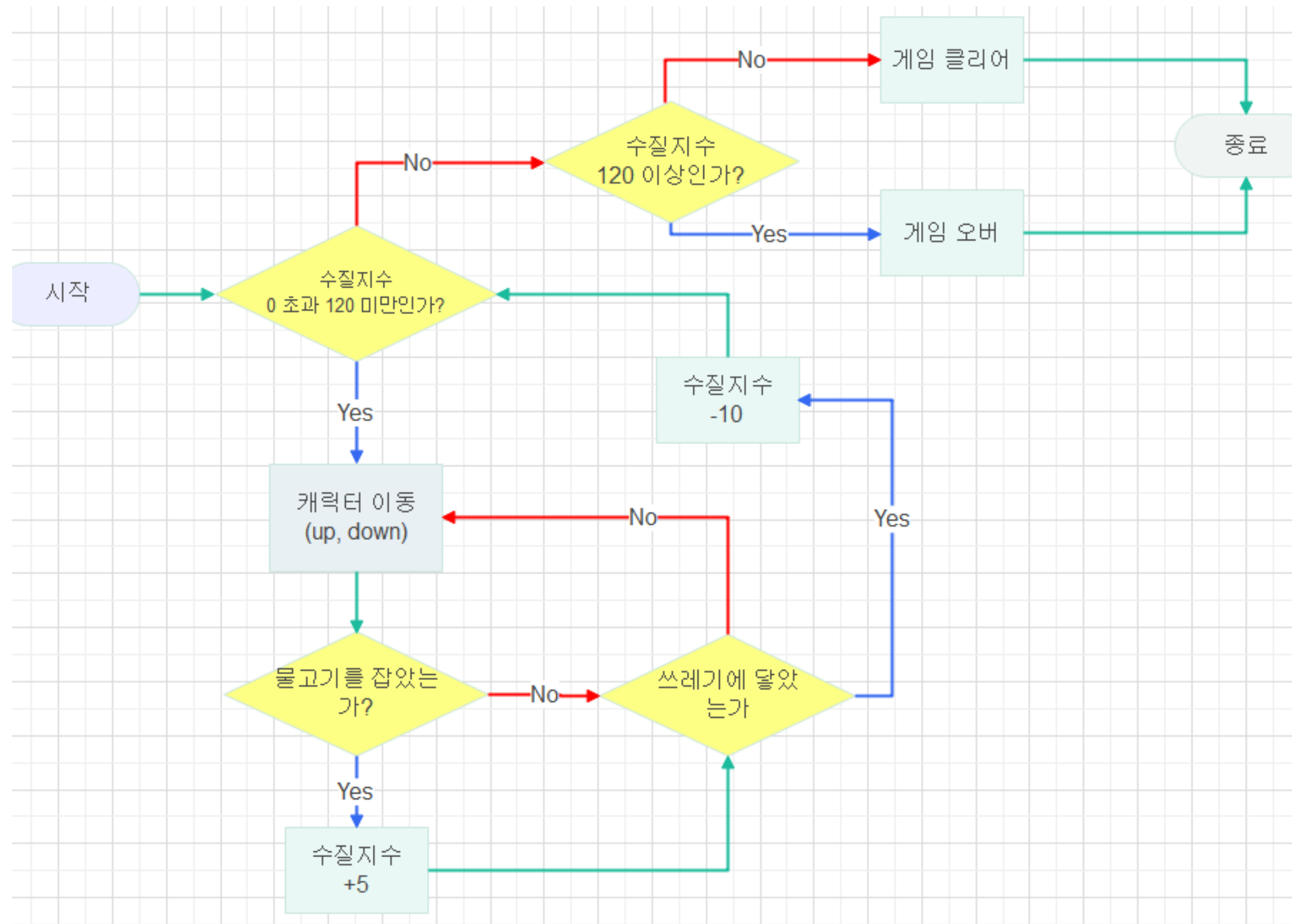
- #26. 게임 플레이 함수

메인

- #27. 시작 화면

4. 소스코드

4.1 게임 소스코드 구조 - 플로우 차트



4. 소스코드

4.2 게임 소스코드 상세

4.2.1 메인 화면

```
9  #1 화면 크기 설정
10 window_width, window_height = 1280, 800
11 window = pygame.display.set_mode((window_width, window_height))
12 pygame.display.set_caption("바다를 지켜라!")
13
```



4.2.2 음악 및 이미지 로드

```
14 #2 음악 사운드 로드
15 background_music = "all_sounds/main_music.mp3"
16 collision_sound = pygame.mixer.Sound('all_sounds/sound2.mp3')
17 collision_sound.set_volume(0.3)
18 pygame.mixer.music.load(background_music)
19
20 #3 메인화면 이미지 로드
21 background_image = pygame.image.load("all_images/background_opening.png")
22 background_image = pygame.transform.scale(background_image, (window_width, window_height))
23
24 #4 게임설명 이미지 로드
25 instruction_image = pygame.image.load('all_images/guide.png')
26 instruction_image = pygame.transform.scale(instruction_image, (window_width, window_height))
27
28 #5 메인화면 버튼 이미지 로드
29 button1_image = pygame.image.load("all_images/game_start.png")
30 button2_image = pygame.image.load("all_images/game_guide.png")
31 button3_image = pygame.image.load("all_images/game_exit.png")
32
33 #6 화면 나가기 이미지 로드
34 arrow1_image = pygame.image.load("all_images/arrow1.png")
35 arrow1_image = pygame.transform.scale(arrow1_image, (48, 40))
36
37 #7 메인 게임 화면 로드
38 gbackground_image = pygame.image.load('all_images/background_ongame.png')
39 gbackground_image = pygame.transform.scale(gbackground_image, (window_width, window_height))
40
41 #8 플레이어 이미지 로드
42 player_image = pygame.image.load('all_images/fishpole_down.png')
43 player_rect = player_image.get_rect()
44 player_rect.x = 50
45 player_rect.y = window_height//2 - player_rect.height//2
46 player_speed = 8
```

```
48 #9 물고기 이미지 로드
49 fish_list = []
50 for i in range(1,8):
51     fish_image = pygame.image.load(f'all_images/fish{i}.png')
52     fish_list.append(fish_image)
53
54 #10 물고기 기본 위치 설정
55 fish_rect = fish_image.get_rect()
56 fish_rect.x = 1300
57 fish_rect.y = random.randint(60,700)
58 fish_speed = 3
59
60 #11 쓰레기 이미지 로드
61 trash_list = []
62 for i in range(1,8):
63     trash_image = pygame.image.load(f'all_images/trash{i}.png')
64     trash_list.append(trash_image)
65
66 #12 쓰레기 기본 위치 설정
67 trash_rect = fish_image.get_rect()
68 trash_rect.x = 1300
69 trash_rect.y = random.randint(60,700)
70 trash_speed = 3
71
72 #13 게임 클리어 화면 로드
73 clear_image = pygame.image.load("all_images/clear_game.png")
74 clear_image = pygame.transform.scale(clear_image, (window_width, window_height))
75
76 #14 게임 오버 화면 로드
77 over_image = pygame.image.load("all_images//game_over.png")
78 over_image = pygame.transform.scale(over_image, (window_width, window_height))
79
```

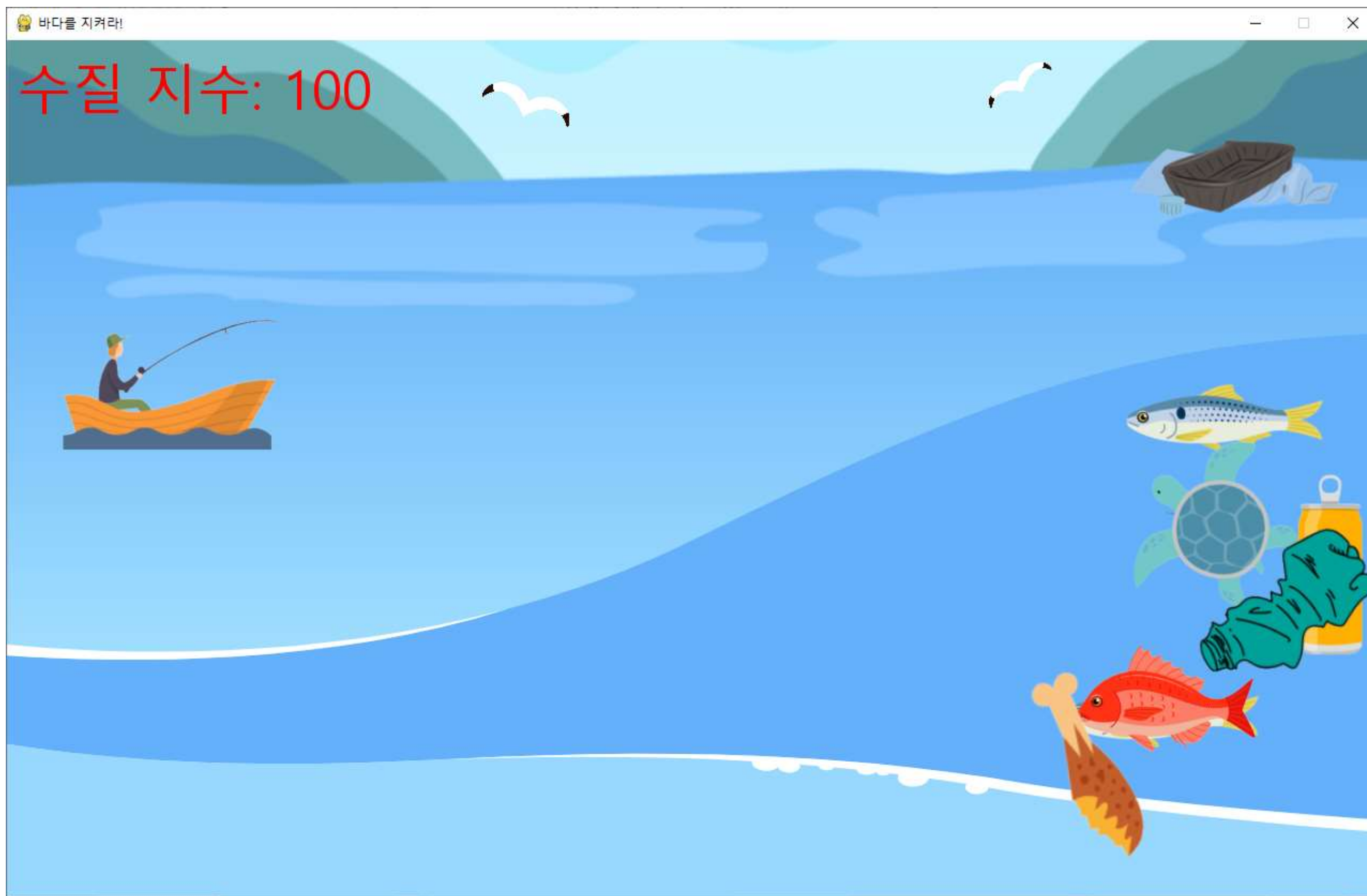
바다를 지켜라

시작

게임설명

종료







arrow1



background_guide1



background_guide2



background_ongame



background_opening



clear_game



fish1



fish2



fish3



fish4



fish5



fish6



fish7



fish8



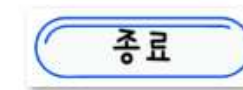
fishpole



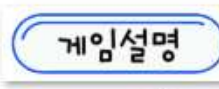
fishpole_down



fishpole_up



game_exit



game_guide



game_over



game_start



guide



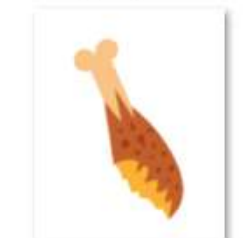
page



shark



trash1



trash2



trash3



trash4



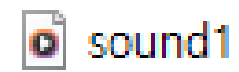
trash5



trash6



trash7



4.2.3 이미지 그리기

```
80 #15 게임 클리어 및 오버 화면 등장 함수 정의
81 def draw_game_over_screen():
82     window.blit(over_image, (0, 0))
83 def draw_clear_screen():
84     window.blit(clear_image, (0, 0))
85
86 #16 메인화면 버튼 기본 값 설정
87 button_width, button_height = 248, 85
88 button_x, button_y = (window_width - button_width) // 2, window_height // 2
89
```

```
90 #17 메인화면 버튼 클래스 생성
91 class Button:
92     def __init__(self, x, y, width, height, image):
93         self.rect = pygame.Rect(x, y, width, height)
94         self.image = image
95         self.enabled = True
96         self.visible = True
97
98     # 게임 화면에 버튼 그리기 함수
99     def draw(self, window):
100         if self.visible:
101             window.blit(self.image, self.rect)
102
103     # 사용자의 버튼클릭 감지 함수
104     def is_clicked(self, pos):
105         return self.enabled and self.visible and self.rect.collidepoint(pos)
106
107     # 버튼 기능 비활성화 함수
108     def disable(self):
109         self.enabled = False
110
111     # 버튼 숨기기 함수
112     def hide(self):
113         self.visible = False
114
115     # 버튼 활성화 함수
116     def enable(self):
117         self.enabled = True
118
119     # 버튼 등장 함수
120     def show(self):
121         self.visible = True
122
```

4.2.3 이미지 그리기

```
123 #18 Button 클래스를 활용하여 버튼 객체 생성
124 button1 = Button(button_x, button_y, button_width, button_height, button1_image)
125 button2 = Button(button_x, button_y + 95, button_width, button_height, button2_image)
126 button3 = Button(button_x, button_y + 190, button_width, button_height, button3_image)
127 buttons = [button1, button2, button3]
128
129 #19 화면 나가기 버튼 객체 생성
130 arrow1 = Button(1220, 0, 48, 40, arrow1_image)
131
132 #20 버튼이미지 3개 그리는 함수
133 def draw_buttons():
134     for button in buttons:
135         button.draw(window)
136
137 #21 메인화면과 버튼을 그리는 함수
138 def draw_opening_screen():
139     window.blit(background_image, (0, 0))
140     draw_buttons()
141
142 #22 설명화면과 나가기 버튼을 그리는 함수
143 def draw_instruction_screen():
144     window.blit(instruction_image, (0, 0))
145     arrow1.draw(window)
146
147
148 previous_screen = [] # 설명화면으로 넘어간 경우 일정 값을 저장하기 위한 리스트 설정
149
```

4.2.4 기능 설정 함수

```
150 #23 메인화면에서 버튼을 클릭했을 때 작동하는 함수
151 def handle_button_click(pos):
152     for button in buttons:
153         if button.is_clicked(pos):
154             if button == button1: #첫번째 게임시작 버튼을 클릭하면 게임 작동 함수 불러오기
155                 game_start()
156             elif button == button2: #두번째 게임 설명 버튼을 클릭하면 설명 페이지 불러오기
157                 draw_instruction_screen()
158                 previous_screen.append("opening") #previous_screen 리스트에 opening라는 문자열 추
159                 for button in buttons: #메인화면에 있던 3개 버튼의 이미지를 숨기기
160                     button.hide()
161             elif button == button3: #세번째 게임 종료 버튼을 누르면 게임 종료
162                 pygame.time.delay(30)
163                 pygame.quit()
164                 sys.exit()
165
```

```
166 #24 화면 나가기 버튼 눌렀을 때 작동하는 함수
167 ~ def handle_arrow_click(pos): #설명화면 나가기 버튼을 누르면 이전 메인화면 불러오는 함수
168 ~     if arrow1.is_clicked(pos):
169 ~         if previous_screen: #previous_screen "opening"이라는 값이 있으면 해당 값을 삭제
170             previous = previous_screen.pop()
171 ~         if previous == "opening":
172             draw_opening_screen()
173 ~         for button in buttons:
174             button.show()
175
```


4.2.4 기능 설정 함수

```
189 #26 게임 플레이 함수
190 def game_start():
191     wqi = 100
192     fish_instances = []
193     trash_instances = []
194
195     # 물고기와 쓰레기 리스트에 랜덤으로 이미지 및 기본 값 설정
196     for _ in range(4):
197         fish_image = random.choice(fish_list)
198         fish_rect = fish_image.get_rect() #get._rect 이미지의 경계 사각형 객체를 반환하는 메서드
199         fish_rect.x = window_width
200         fish_rect.y = random.randint(60, 700)
201         fish_speed = random.uniform(2, 6)
202         fish_instances.append((fish_image, fish_rect, fish_speed)) #fish_instances 리스트 안에 3가지 요소를 튜플 형태로 저장
203
204         trash_image = random.choice(trash_list)
205         trash_rect = trash_image.get_rect()
206         trash_rect.x = window_width
207         trash_rect.y = random.randint(60, 700)
208         trash_speed = random.uniform(2, 6)
209         trash_instances.append((trash_image, trash_rect, trash_speed))
210
211
212     pygame.mixer.music.play(-1) #배경 음악을 반복 재생하는 메서드
213
214     game_over = False
```

4.2.4 기능 설정 함수

```
217  while not game_over:
218      for event in pygame.event.get(): #사용자 입력을 감지하고 게임 상태를 업데이트하거나 처리하는 등의 작업을 수행
219          if event.type == pygame.QUIT:
220              pygame.mixer.music.stop()
221              pygame.quit()
222              sys.exit()
223  if wqi >=120: #WQI(수질오염지수)가 120이상이면 게임 오버 화면이 나오고 다시 메인 화면으로 이동
224      window.blit(over_image, (0, 0))
225      pygame.display.flip()
226      pygame.time.delay(6000)
227      game_over = True
228
229  elif wqi <=0: #WQI(수질오염지수)가 0이하가 되면 게임 클리어 화면이 나오고 다시 메인 화면으로 이동
230      window.blit(clear_image, (0, 0))
231      pygame.display.flip()
232      pygame.time.delay(6000)
233      game_over = True
234
235
236  if wqi <= 60: #플레이어가 WQI(수질오염지수)를 60으로 낮췄을 경우 속도가 빨라지는 필살기 적용
237      player_speed *= 4
238
239  elif wqi>=115:#플레이어가 WQI(수질오염지수)를 115이상으로 높였을 경우 속도를 낮추는 패널티 적용
240      player_speed *=0.35
```

4.2.4 기능 설정 함수

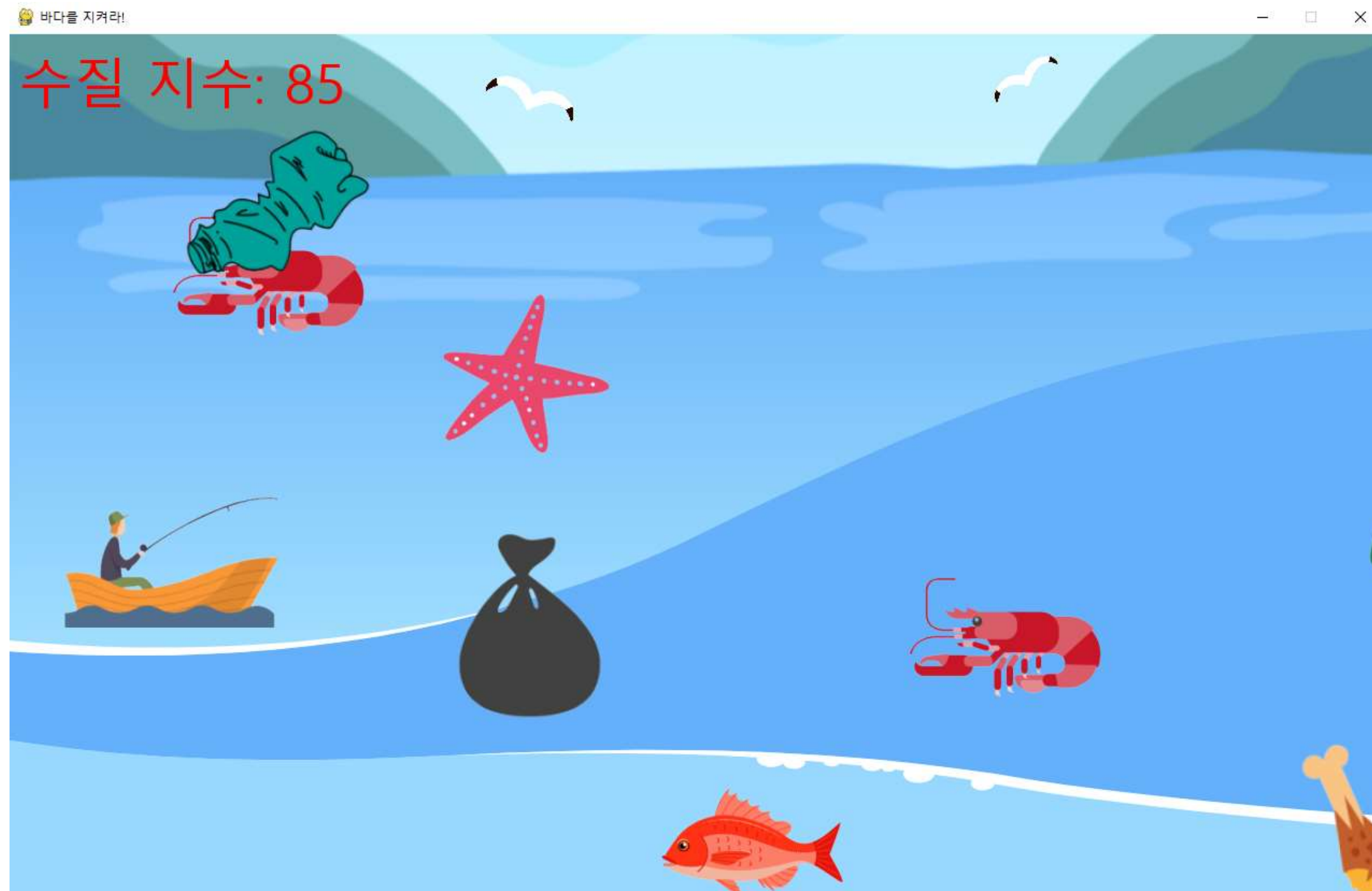
```
242 # 플레이어 키보드 움직임 설정
243 player_speed = 10
244 keys = pygame.key.get_pressed() #pygame.key.get_pressed는 현재 눌러져 있는 키들에 대한 상태를 리스트 형태로 반환
245 if keys[pygame.K_UP]: #키보드 상향키를 눌렀을 경우 아래와 같이 동작
246     player_rect.y -= player_speed #객체의 y 좌표를 player_speed만큼 감소시켜 플레이어의 위치를 위로 이동
247     if player_rect.y < 50: #플레이어화면 크기에 맞게 플레이어의 위치를 제한
248         player_rect.y = 50
249 elif keys[pygame.K_DOWN]: #키보드 하향키를 눌렀을 경우 아래와 같이 동작
250     player_rect.y += player_speed #객체의 y 좌표를 player_speed만큼 증가시켜 플레이어의 위치를 아래로 이동
251     if player_rect.y > 800 - player_rect.height:
252         player_rect.y = 800 - player_rect.height
253
254 # 물고기 움직임 설정
255 for fish_image, fish_rect, fish_speed in fish_instances: #fish_instances 리스트에 튜플 형태로 저장된 3가지 요소를 반복문을 통해 물고기의 움직임을 지정
256     fish_rect.x -= fish_speed #물고기가 화면 오른쪽에 왼쪽으로 이동
257     if fish_rect.right < 0: #물고기가 화면 밖을 벗어나면 다시 오른쪽 화면에서 등장
258         fish_rect.y = random.randint(60, 700)
259         fish_rect.x = window_width
260
261 # 쓰레기 움직임 설정
262 for trash_image, trash_rect, trash_speed in trash_instances:
263     trash_rect.x -= trash_speed
264     if trash_rect.right < 0:
265         trash_rect.y = random.randint(60, 700)
266         trash_rect.x = window_width
267
268 # 화면에 있는 물고기와 쓰레기 이미지들을 리스트 변수에 저장
269 fish_instances = [(image, rect, speed) for image, rect, speed in fish_instances if rect.right >= 0]
270 trash_instances = [(image, rect, speed) for image, rect, speed in trash_instances if rect.right >= 0]
271
```

4.2.4 기능 설정 함수

```
272 # 플레이어와 물고기와 충돌한 경우 수질오염지수를 5점 증가시키고 해당 이미지 제거후 다시 등장
273 for fish_image, fish_rect, _ in fish_instances[:]: # Use [:] to make a copy of the list
274     if player_rect.colliderect(fish_rect):
275         wqi += 5
276         fish_instances.remove((fish_image, fish_rect, _))
277         fish_rect.y = random.randint(60, 700)
278         fish_rect.x = window_width
279         fish_instances.append((fish_image, fish_rect, fish_speed))
280         collision_sound.play()
281
282 # 플레이어와 쓰레기가 충돌한 경우 수질오염지수를 10점 감소시키고 이미지 제거 후 다시 등장
283 for trash_image, trash_rect, _ in trash_instances[:]: # Use [:] to make a copy of the list
284     if player_rect.colliderect(trash_rect): #
285         wqi -= 10
286         trash_instances.remove((trash_image, trash_rect, _))
287         trash_rect.y = random.randint(60, 700)
288         trash_rect.x = window_width
289         trash_instances.append((trash_image, trash_rect, trash_speed))
290         collision_sound.play()
291
292 # 플레이 배경화면과 게임 이미지들을 그리는 함수
293 window.blit(gbackground_image, (0, 0))
294 window.blit(player_image, player_rect)
295 for fish_image, fish_rect, _ in fish_instances:
296     window.blit(fish_image, fish_rect)
297 for trash_image, trash_rect, _ in trash_instances:
298     window.blit(trash_image, trash_rect)
```

```
300 # WQI(수질오염지수)를 화면에 표시
301 wqi_color = get_wqi_color(wqi)
302 font = pygame.font.SysFont("malgun Gothic", 50) #pygame.font.Font(None, 50)
303 wqi_text = font.render(f"수질 지수: {wqi}", True, wqi_color)
304 window.blit(wqi_text, (10, 10))
305 pygame.display.flip()
```


4.2.4 기능 설정 함수



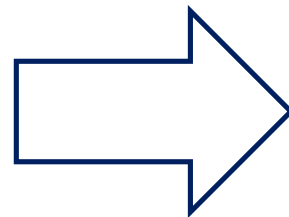
4.2.5 기타 변수 설정

```
176 #25 WQI(수질오염지수)에 따른 색상 변환 함수
177 def get_wqi_color(wqi):
178     if wqi < 24:
179         return (0, 0, 255)
180     elif 24 <= wqi < 34:
181         return (135, 206, 250)
182     elif 34 <= wqi < 47:
183         return (255, 255, 0)
184     elif 47 <= wqi < 60:
185         return (255, 165, 0)
186     elif wqi >= 60 :
187         return (255,0,0)
```



4.2.6 메인 루프

```
308 #27 메인 시작화면
309
310 while True:
311
312     for event in pygame.event.get(): # pygame의 모든 이벤트(사용자 입력)을 저장하는 버퍼를 가져옴
313         # 창을 닫기 위해 종료하는 이벤트가 있는지 확인
314         # pygame.QUIT : 윈도우의 닫기버튼(x버튼)을 누르는 이벤트
315         if event.type == pygame.QUIT:
316             pygame.mixer.music.stop()
317             # pygame 종료
318             pygame.quit()
319             # 파이썬 프로그램 종료
320             sys.exit()
321         # 마우스 버튼을 누르는 이벤트 발생 확인
322         elif event.type == pygame.MOUSEBUTTONDOWN:
323             if event.button == True: # 마우스의 버튼 값을 확인
324                 pos = pygame.mouse.get_pos() # 마우스의 커서의 위치를 가져와서 pos 변수에 저장
325
326                 if previous_screen: #설명페이지에서 버튼 버튼 클릭이 인식되고 previous_screen값이 True으로 된 경우
327                     handle_arrow_click(pos) # 이전 페이지가 사라지고(.pop) 메인 페이지와 버튼 등장
328                 else:
329                     handle_button_click(pos) #버튼 클릭 화면이 메인 페이지일 경우, 3가지 버튼이 동작하는 함수 호출
330
331
332     if previous_screen:
333         draw_instruction_screen()
334     else:
335         draw_opening_screen() #사용자의 키보드값 입력 상관없이 기본 메인화면 설정
336
337     pygame.display.flip() #화면을 업데이트하는 역할
338     pygame.time.Clock().tick(120) #화면이 좀 더 부드럽게 움직일 수 있도록 게임 루프가 초당 120번 실행되도록 설정
```



6. 마케팅 활용 방안

6. 마케팅 활용 방안

- 남녀노소 사용자 연령층 제한이 없어 범국민적 **힐링게임**
- **어린이 환경보호** 교육 콘텐츠 활용
- 해양 환경 보호 인식 개선 및 **공익 캠페인 효과** 최대화
- 게임 플레이 맛보기 시연, **체험 기회 제공** 및 홍보 극대화

감사합니다!