Busan science high school

2023 Ocean ICT Festival 2023 BOIF

B 42

QR 코드 영역
QR 삽입 후
테두리 삭제

Youtube 영상 QR

옥시벤존 농도 모니터링 프로그램을 통한 해양생태계 보호

조수방바다홍수경보
3203 조수현 3306 홍수민

[해양 문제]

선크림의 주성분 (옥시벤존, 옥티노세이트) :
산호의 DNA 손상 유도해, 어린 산호의 기형 생성 & 산호 자체 껍질의 성장 막아 즉게 함

[융합 분야]

정보과학 + **생명과학** + **화학**

1. 탐구 목적

2022 Ocean ICT Youtube 영상 →

[2022 Ocean ICT]

해수의 옥시벤존 농도를 측정해 특정 값 이상이 되면 경고를 울리고 옥시벤존 분해 물질을 투여하는 코드를 작성하여 옥시벤존의 농도를 줄이는 탐구를 진행함.

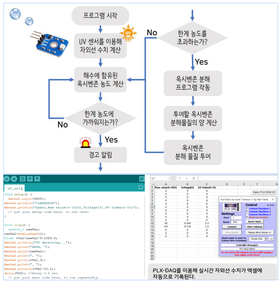
[2023 Ocean ICT]

하지만, 해수의 옥시벤존을 직접적으로 분해시키는 방법은 매우 일시적인 효과를 가지며, 가성비와 효율이 좋지 못하다는 단점이 있었다.
따라서, 이번에는 접근법에 변화를 주어 바다로 가는 사람들이나 해변에서 활동하는 사람들을 대상으로 옥시벤존과 옥티노세이트가 함유된 선크림을 사용하지 않도록 경고하거나 대체 제품을 추천하는 프로그램을 제작하고자 하였다.

2. 알고리즘

- **옥시벤존 분해 물질(해조류(미생물))을 투입하여 옥시벤존이 산호의 DNA를 손상시키는 농도에 이르지 못하게 한다.**
- **여러 개의 센서를 동시에 사용하여 다양한 위치에서 옥시벤존 농도를 모니터링하는 시스템을 구축할 수 있다. 이를 통해 여러 지역의 산호초를 동시에 관찰하고, 옥시벤존 농도의 변화를 통합적으로 파악할 수 있다.**
- **옥시벤존과 옥티노세이트가 함유된 선크림을 사용하지 않도록 경고하거나 대체 제품을 추천하는 앱 또는 웹 애플리케이션을 개발할 수 있다. 앱으로 사용자가 자신이 사용하는 제품의 성분을 확인하고, 친환경적이고 해로운 성분을 포함하지 않은 대안을 제공할 수 있다.**

옥시벤존 분해 물질 투입 알고리즘



산호초에 유해한 화학 물질

옥시벤존
일부 연구에 따르면, 옥시벤존은 산호초의 민감도를 증가시키고, 산호초와 해조류의 생장을 저해하는 요인으로 작용함

옥시노세이트
해수로 배출되어 산호초 생태계에 영향을 미치고, 산호초의 건강을 해칠 수 있는 화학 물질로 알려져 있음

자연 친화적인 성분

아연 산화물
물리적 자외선 차단제, 자연에서 추출되거나 합성된 성분
피부 표면에 미세한 보호막을 형성하여 자외선을 반사함

티타늄 산화물
물리적 자외선 차단제
자외선을 반사하여 피부를 보호함

- **순수 아연 선크림**
- **아연 산화물과 티타늄 산화물 혼합 선크림**

All Good - "All Good Mineral Sunscreen"
Badger - "Badger Clear Zinc Sunscreen"
Blue Lizard - "Blue Lizard Australian Sunscreen"
Thinksport - "Thinksport Safe Sunscreen"

3. 연구 과정 및 코드 작성

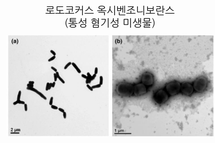
[옥시벤존 농도 모니터링]

통합 관리 시스템

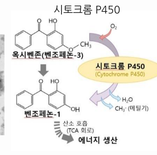
```
import time
from random import uniform

# 여러 개의 센서를 가정하고, 센서별의 데이터를 관리하기 위한 딕셔너리 생성
sensors = {
    "sensor1": {
        "location": "Location 1",
        "pin": 1,
        "threshold": 10.0,
        "last_reading": 0.0
    },
    "sensor2": {
        "location": "Location 2",
        "pin": 2,
        "threshold": 15.0,
        "last_reading": 0.0
    },
    "sensor3": {
        "location": "Location 3",
        "pin": 3,
        "threshold": 12.0,
        "last_reading": 0.0
    }
}

def read_sensor_data(sensor):
    # 센서에서 데이터를 읽는 함수 (실제로는 하드 코딩을 사용하여 가정)
    sensor_value = uniform(0.0, 20.0)
    return sensor_value

def send_alert(sensor_name, location, value):
    # 경고 알림을 보냄 (실제로는 콘솔에 출력하는 것으로 가정)
    print(f"ALERT! Sensor {sensor_name} at {location} detected a value of {value} above the threshold!")

# 주기적으로 센서 데이터를 읽고 처리하는 메인 루프
while True:
    for sensor_name, sensor_info in sensors.items():
        location = sensor_info["location"]
        pin = sensor_info["pin"]
        threshold = sensor_info["threshold"]
        last_reading = sensor_info["last_reading"]

        # 센서에서 데이터를 읽어옴
        sensor_value = read_sensor_data(pin)

        # 이전 값과 비교하여 경고 알림을 보냄
        if sensor_value > threshold and sensor_value > last_reading:
            send_alert(sensor_name, location, sensor_value)

        # 이전 값을 업데이트
        sensors[sensor_name]["last_reading"] = sensor_value

    # 일정 시간 간격으로 반복 실행 (여기서는 5초로 설정)
    time.sleep(5)
```

각 센서에 대한 정보를 딕셔너리로 관리하고, 주기적으로 각 센서에서 데이터를 읽어와서 경고 알림을 보내는 기능을 구현한다.

로도코커스 옥시벤조니보란스
(통성 혐기성 미생물)



시토크롬 P450



[대체 제품 프로그램]

```
from flask import Flask, request, jsonify

app = Flask(__name__)

# 대체 제품 데이터베이스
alternative_products = {
    "product1": {
        "name": "Alternative Product 1",
        "description": "This is an alternative product without oxybenzone and octinoxate."
    },
    "product2": {
        "name": "Alternative Product 2",
        "description": "This is another alternative product that is environmentally friendly."
    }
}

@app.route("/check_product", methods=["POST"])
def check_product():
    data = request.get_json()
    product_name = data["product_name"]

    # 사용자가 제품의 성분을 확인하고 경고 메시지를 반환하는 로직
    if is_harmful(product_name):
        message = "Warning! The product contains oxybenzone and octinoxate."
    else:
        message = "The product is safe to use."

    return jsonify({"message": message})
```

```
@app.route('/recommend_products', methods=['GET'])
def recommend_products():
    # 대체 제품을 추천하는 로직
    recommended_products = []
    for product_id, product_info in alternative_products.items():
        recommended_products.append({
            "id": product_id,
            "name": product_info["name"],
            "description": product_info["description"]
        })

    return jsonify({"recommended_products": recommended_products})

def is_harmful(product_name):
    # 성분을 확인하고 해로운 성분을 포함하는지 판별하는 로직
    if product_name == "Harmful Product":
        return True
    else:
        return False

if __name__ == "__main__":
    app.run()
```

Flask 웹 프레임워크를 사용하여 RESTful API를 구현한다.
'/check_product' 엔드포인트는 제품의 성분을 확인하고 경고 메시지를 반환하며, '/recommend_products' 엔드포인트는 대체 제품을 추천한다.

선크림 데이터 베이스

```
# 데이터베이스 선크림 정보 (이름, 성분 포함 여부)
sunscreen_database = [
    {"name": "선크림A", "ingredients": ["옥시벤존"]},
    {"name": "선크림B", "ingredients": ["옥시노세이트"]},
    {"name": "선크림C", "ingredients": ["옥시벤존", "옥시노세이트"]},
    # 다른 선크림 정보도 추가 가능
]

# 자연 친화적인 선크림 목록
natural_sunscreen_recommendations = [
    {"name": "자연 선크림1", "ingredients": ["아연 산화물", "티타늄 산화물"]},
    {"name": "자연 선크림2", "ingredients": ["아연 산화물"]},
    {"name": "자연 선크림3", "ingredients": ["친환경 산화물"]},
    # 다른 자연 친화적인 선크림 정보도 추가 가능
]

def recommend_natural_sunscreen():
    user_input = input("사용하려는 선크림 이름을 입력해주세요: ")

    matching_sunscreen = None
    for sunscreen in sunscreen_database:
        if user_input == sunscreen["name"]:
            matching_sunscreen = sunscreen
            break

    if matching_sunscreen is None:
        print("입력하신 선크림 이름이 데이터베이스에 없습니다. 다시 입력해주세요.")
        recommend_natural_sunscreen()
    else:
        if "옥시벤존" in matching_sunscreen['ingredients'] or '옥시노세이트' in matching_sunscreen['ingredients']:
            print("경고: 해당 선크림에는 함께 성분이 포함되어 있을 수 있습니다.")
            print("환경 보호와 피부 건강을 고려하여 자연 친화적인 성분을 포함한 선크림을 추천합니다:")
            for natural_sunscreen in natural_sunscreen_recommendations:
                print(f"- {natural_sunscreen['name']}")
        else:
            print("해당 선크림은 함께 성분을 포함하지 않습니다. 바다에서 즐거운 시간을 보내세요!")

# 함수 호출
recommend_natural_sunscreen()
```

+ 환경 인식 개선

```
def recommend_sunscreen():
    used_sunscreen = input('이전 선크림을 사용하셨나요? (예/아니오): ')

    if used_sunscreen.lower() == '예':
        sunscreen_name = input('어떤 종류의 선크림을 사용하셨나요? ')
        matching_sunscreen = None
        for sunscreen in sunscreen_database:
            if sunscreen_name == sunscreen['name']:
                matching_sunscreen = sunscreen
                break
        if matching_sunscreen is None:
            print('선택한 선크림을 찾을 수 없습니다.')
        else:
            print(f'{sunscreen_name} 선크림을 사용하셨군요.')
            print('사용하신 선크림을 바다에서 물로 충분히 씻겨서고, 바다 안전 지침을 따르시기 바랍니다.')
            print('바다 안전 지침 및 환경 관련 경고 알림을 확인하세요.')
            # 바다 관련 지침과 환경 관련 경고 알림 보내주는 코드 추가
    else:
        print('이전 선크림을 사용하지 않으셨군요. 어떤 종류의 선크림을 고려하고 계신가요?')
        print('선크림이 바다와 환경에는 상황 좋아가 위해 노력하시고, 사용 후에는 남아있는 선크림을 물로 제거해주세요.')
        print('자연 친화적인 성분이 포함된 선크림을 추천합니다.')
        for natural_sunscreen in natural_sunscreen_recommendations:
            print(f'- {natural_sunscreen['name']}')
        print('바다 안전 지침 및 환경 관련 경고 알림을 확인하세요.')
        # 자연 친화적인 선크림 추천과 관련 경고 알림 보내주는 코드 추가

    # 바다 안전 지침과 환경 관련 경고 알림을 보내주는 코드
    print('바다 안전 지침을 준수하고 환경 보호에 동참해주세요.')
    print('바다 안전 지침 사항을 확인하려면 아래 링크를 참고하세요.')
    print('환경 보호와 관련한 영상을 시청하려면 아래 링크를 확인하세요.')

# 함수 호출
recommend_sunscreen()
```

4. 기대효과

1. 조기 경고와 대응 능력 강화

센서를 이용한 실시간 모니터링을 통해 환경 변화나 유해 물질 농도의 증가를 빠르게 감지하고 조기에 대응할 수 있다. 이로써 유해한 사건이 확대되거나 국면이 악화되는 것을 막을 수 있을 것이다.

2. 생태계 변화 관찰

모니터링 시스템은 해양 생태계의 변화를 지속적으로 관찰하고 분석할 수 있는 기회를 제공한다. 이를 통해 환경 변화에 따른 생태계의 변화를 이해하고 예측할 수 있다.

3. 자료 기반 정책 수립

모니터링 결과로 얻은 데이터를 기반으로 정책을 수립하고 실행할 수 있다. 실제 자료를 가지고 문제를 분석하고 이를 해결하는 정책을 만들 수 있을 것이다.

4. 환경 보호 의식 제고

모니터링 데이터를 통해 해양 환경의 변화와 위험성을 시각적으로 보여줄 수 있다. 이로 인해 사람들의 환경 보호 의식이 높아지고 지속 가능한 환경 관리에 대한 관심이 증가할 수 있을 것이다.

5. 참고 문헌

- 자외선차단제 사용 '옥시벤존' 분해 미생물 발견 (국립낙동강생물자원관)
- 선블록과 백화현상 - 약학정보원
- Archives of Environmental Contamination and Toxicology (2015)
- 사이언스지 (2022. 05)