스터디 17 차시
ROS 정기평가 대비 예상 문제

홍송은

1. 다음 중 ROS2 의 DDS 미들웨어에 대한 설명으로 옳은 것을 모두 고르시오.
   a. DDS 는 퍼블리셔와 서브스크라이버가 명시적으로 연결되어야 한다.
      ➡ DDS 는 퍼블리셔/서브스크라이버 간 명시적 연결 필요 없음 (브로커리스 구조)
   b. ROS2 에서 DDS 를 사용하면 마스터 노드 없이도 동적 노드 검색이 가능하다.
   c. DDS 는 데이터 중심 설계를 기반으로 QoS 설정이 불가능하다.
      ➡ DDS 는 강력한 QoS 정책을 지원함
   d. ROS2 는 DDS 를 통해 다양한 운영체제와 언어에서 상호 운용이 가능하다.
   e. DDS 는 TCP 기반 통신만 지원하며, 멀티캐스트는 사용할 수 없다.
      ➡ DDS 는 UDP 멀티캐스트도 지원함

2. ROS2 의 QoS 정책 중, 노드가 일정 시간마다 메시지를 발행하지 않으면 상태를 감지해 오류를 발생시키는 설정은?
   a. Deadline
   b. Reliability
   c. Durability
   d. Liveliness
   e. Lifespan

3. 다음 명령은 turtlesim 노드를 파라미터 파일로 실행하는 명령이다. 빈칸 (A)에 들어갈 알맞은 옵션은?

```
ros2 run turtlesim turtlesim_node (A) ./params.yaml
```

   a. --config-file
   b. --ros-params
   c. --launch-config
   d. --ros-args --params-file
   e. --yaml-input

4. ROS2 에서 Action 통신의 특징으로 올바른 설명을 모두 고르시오.
   a. Action 은 비동기식으로 결과를 기다릴 수 있다.
   b. Action 은 단순 토픽보다 높은 주기성과 실시간성이 요구되는 서비스에 적합하다.
   c. Action 은 Feedback, Goal, Result 세 가지 메시지 구조를 가진다.
   d. Action 은 QoS 정책을 적용할 수 없다.
      ➡ QoS 적용 가능함
   e. Action 은 항상 Node 간 RPC 방식으로 동작한다.
      ➡ Action 은 RPC 기반이 아닌 토픽/서비스 혼합 구조

5. 다음 중 ROS2 에서 서비스와 액션의 차이에 대한 설명으로 가장 올바른 것은?
    a. 서비스는 비동기이고 액션은 동기식이다.
    b. 액션은 복잡한 트랜잭션을 실행할 수 없다.
    c. 서비스는 결과가 즉시 반환되는 작업에 적합하고, 액션은 시간이 오래 걸리는 작업에 적합하다.
    d. 액션은 QoS 를 지원하지 않는다.
    e. 서비스는 feedback 메시지를 포함할 수 있다.

6. 아래 명령어의 실행 결과로 알맞은 것을 고르시오.

```
ros2 run turtlesim turtlesim_node --ros-args -r __ns:=/robot1
```

    a. /robot1/turtlesim_node 라는 이름의 노드를 생성한다.
    b. turtlesim_node 가 /robot1 토픽을 삭제한다.
    c. /turtlesim_node 의 namespace 가 /robot1 으로 재매핑된다.
    d. ros2 도메인 ID 를 /robot1 로 고정한다.
    e. 별칭(robot1)을 설정해 alias 실행을 가능하게 한다.

7. 다음은 ROS2 에서 동일한 talker 노드를 두 번 실행하고, 각 노드가 다른 리스너와 연결되도록 구성하려는 상황이다. 각 노드는 다음과 같이 실행되었다:

```
ros2 run demo_nodes_cpp talker --ros-args -r chatter:=/robot1/chatter --namespace robot1

ros2 run demo_nodes_cpp talker --ros-args -r chatter:=/robot2/chatter --namespace robot2

ros2 run demo_nodes_cpp listener --ros-args -r chatter:=/robot1/chatter --namespace robot1

ros2 run demo_nodes_cpp listener --ros-args -r chatter:=/robot2/chatter --namespace robot2
```

    위 네 개의 노드 실행 결과, 다음 설명 중 가장 정확한 것은?

    a. 두 talker 노드는 동일한 토픽 /chatter 를 사용하고 있기 때문에 충돌이 발생한다.
       ➡ 토픽 충돌 없음
    b. robot1 의 listener 는 robot2 의 talker 에게서도 메시지를 수신한다.
       ➡ namespace 격리되어 수신 못 함
    c. 네임스페이스와 리맵이 독립적으로 설정되어 각 노드 쌍(robot1, robot2)은 분리된 통신 경로를 사용한다.
    d. 리맵핑을 사용할 경우 --namespace 옵션은 무시되므로 모든 노드가 /chatter 로 연결된다.
       ➡ remap 과 namespace 는 독립 작동
    e. ROS2 에서는 하나의 talker 만 있어야 하며, 두 개를 실행하면 ros2 run 이 실패한다.
       ➡ 여러 talker 가능

8. 다음 중 ROS2 에서 Python 패키지를 구성할 때, 실행 가능한 패키지로 인식되기 위해 반드시 필요한 파일은?
    a. setup.py 또는 setup.cfg
    b. requirements.txt
    c. build.bash
    d. install.yaml
    e. run_ros2.py

9. ROS2 에서 setup.py, setup.cfg 의 특징에 대한 설명 중 옳은 것을 모두 고르시오.
    a. setup.py 는 선언형 설정이며 수정이 간단하다.
        ➡ 선언형은 cfg
    b. setup.cfg 는 동적 계산과 사용자 정의가 가능하다.
        ➡ cfg 는 정적 설정
    c. setup.py 는 프로그래밍 방식으로 설정을 구성한다.
    d. 현대 Python 패키징은 setup.cfg 기반 구성을 선호한다.
    e. setup.cfg 는 CMake 기반 패키지에만 사용된다.
        ➡ setup.cfg 는 CMake 와 무관 (Python 전용)

10. 사용자는 다음과 같은 .yaml 파라미터 파일을 작성한 뒤, turtlesim_node 에 적용하려 한다:

    turtle_params.yaml

```
turtlesim_node:
  ros__parameters:
    background_r: 255
    background_g: 255
    background_b: 0
```

    그리고 다음과 같이 노드를 실행했다:

```
ros2 run turtlesim turtlesim_node --ros-args --params-file turtle_params.yaml --rem
ap __node:=sim1
```

    위 실행 결과에 대한 설명 중 가장 옳은 것은?

    a. 노드 이름이 sim1 으로 변경되었기 때문에 파라미터 파일에서 turtlesim_node:로 지정된 키는 무시된다.
    b. background 색상은 노란색으로 적용되며, 노드 이름은 sim1 이지만 파라미터는 정상적으로 적용된다.
        ➡ 파라미터 key 는 실행파일(turtlesim_node)기준이며 remap 된 노드 이름(sim1)과는 무관하게 파라미터 적용됨
    c. 파라미터는 적용되지만 리맵핑된 노드 이름을 찾지 못해 경고가 발생한다.
    d. 파라미터 적용을 위해서는 반드시 노드 이름과 파라미터 키가 일치해야 하므로, turtlesim_node: 대신 sim1:이어야 한다.
    e. ROS2 에서는 .yaml 파일을 사용한 파라미터 설정은 launch 파일에서만 가능하고 ros2 run 에서는 무시된다.

11. 다음 중 ROS2 에서 Action 서버가 cancel_goal() 호출 이후 목표 상태를 바꿀 수 있는 조건으로 옳은 것은?
    a. Goal 상태가 EXECUTING 일 때만 cancel_goal()을 호출할 수 있다.
       ➡ ACCEPTED 에서도 cancel 가능
    b. 목표가 ACCEPTED 상태일 경우 cancel_goal()은 실패한다.
       ➡ 실패 아님
    c. cancel_goal()은 서버가 요청을 수락한 경우에만 CANCELING 상태로 전환된다.
    d. cancel_goal()은 클라이언트가 아닌 서버에서 직접 호출해야 한다.
       ➡ 클라이언트가 요청
    e. Goal 상태가 SUCCEEDED 상태로 바뀐 후에도 cancel_goal()은 유효하다.
       ➡ SUCCEEDED 이후는 cancel 불가

12. 다음 명령어로 기록된 BAG 파일 turtle_bag 을 재생할 때, 발생할 수 있는 상황으로 가장 적절한 것은?

```
ros2 bag record -o turtle_bag /turtle1/cmd_vel
...
ros2 bag play turtle_bag
```

    a. 재생 시 토픽 /turtle1/cmd_vel 이 자동 생성되지 않으면 에러가 발생한다.
       ➡ bag 파일은 기록된 토픽 이름과 메세지 형식 그대로 재전송. 토픽이 없어도 메시지는 발행됨
    b. 실행 중이던 turtlesim_node 가 종료되지 않았다면 플레이어가 거부된다.
    c. turtle 의 이동 경로는 항상 정확히 재현된다.
    d. record 는 topic 이름이 아닌 node 이름 기준으로 기록된다.
    e. 기록된 메시지는 cmd_vel 메시지 구조를 기반으로 재전송된다.

13. 아래 구성에서 progress_action_client 가 목표를 전송한 후 발생하는 콜백 함수로 옳은 순서는?

    ① result_callback

    ② send_goal_async

    ③ feedback_callback

    ④ goal_response_callback

    a. 2 → 4 → 3 → 1
    b. 4 → 2 → 1 → 3
    c. 2 → 1 → 4 → 3
    d. 2 → 3 → 4 → 1
    e. 4 → 3 → 1 → 2

14. 사용자는 Fibonacci.action 기반 액션 서버/클라이언트를 구성하고 다음과 같은 시나리오를 수행하였다.
    - 클라이언트는 order=10 인 goal 을 전송
    - 서버는 goal 을 ACCEPTED 상태로 등록하고 실행 시작
    - 피드백을 전송하는 중, 클라이언트가 cancel_goal 요청을 보냄
    - 서버가 cancel 요청을 수락하고 goal 을 CANCELED 상태로 전환
    - 클라이언트는 result 콜백에서 status 가 CANCELED 인 결과를 수신

    위 상황을 기반으로 한 설명 중, 올바른 것만 모두 고르시오.

    a. cancel 요청은 goal 상태가 EXECUTING 일 때도 보낼 수 있다.
    b. CANCELING 은 서버가 cancel 요청을 수락한 직후의 중간 상태이다.
    c. GoalStatus 메시지에서 status 가 CANCELED 일 경우 결과 데이터는 전달되지 않는다.
       ➡ result 는 CANCELED 라도 전달됨
    d. result 콜백은 cancel 요청 여부와 관계없이 항상 호출된다.
    e. cancel 요청이 거부되면 goal 상태는 REJECTED 로 전환된다.
       ➡ 거절되면 상태 유지, REJECTED 는 goal 수락 실패일 때
    f. send_goal_async()는 결과 콜백과 피드백 콜백 모두를 자동 연결한다.
       ➡ 콜백 수동 등록 필요
    g. 하나의 액션 클라이언트는 여러 개의 goal 을 동시에 전송할 수 없다.
       ➡ 여러 goal 동시에 전송 가능

15. Intra-process communication 이 적용되는 조건으로 옳은 것은?
    a. 퍼블리셔와 서브스크라이버가 동일한 토픽을 사용할 경우 항상 활성화된다.
    b. 퍼블리셔와 서브스크라이버가 같은 노드일 때만 동작한다.
    c. 퍼블리셔와 서브스크라이버가 동일 프로세스(PID) 내에 있어야 한다.
       ➡ 동일 PID 내에서만 ROS2 가 zero-copy 방식으로 최적화된 intra-process 통신 수행
    d. image 토픽이나 PointCloud 토픽은 Intra-process 를 지원하지 않는다.
    e. Intra-process 는 모든 ROS2 설정에서 기본값으로 활성화되어 있다.

16. 다음 QoS 설정은 어떤 동작을 의미하는가?

```
QoSProfile(
    reliability=QoSReliabilityPolicy.RELIABLE,
    durability=QoSDurabilityPolicy.TRANSIENT_LOCAL,
    history=QoSHistoryPolicy.KEEP_LAST,
    depth=5
)
```

    a. 신뢰성은 낮지만 메시지 캐싱은 강력하게 수행된다.
    b. 메시지가 누락될 가능성이 높다.
    c. 구독자가 늦게 붙더라도 이전 메시지를 받을 수 있다.
       ➡ TRANSIENT_LOCAL + RELIABLE
    d. publisher 와 subscriber 가 동시에 붙어야만 메시지를 받을 수 있다.
    e. 메시지는 5 개까지는 저장되지만 재전송은 되지 않는다.

17. 다음 ros2 lifecycle 명령어 시퀀스 실행 결과로 노드가 최종적으로 도달하는 상태는?

```
ros2 lifecycle set /my_lifecycle_node configure
ros2 lifecycle set /my_lifecycle_node activate
ros2 lifecycle set /my_lifecycle_node deactivate
ros2 lifecycle set /my_lifecycle_node cleanup
➡ 노드를 unconfigured 상태로 초기화함
```

a. unconfigured
b. configured
c. inactive
d. active
e. finalized

18. ROS2 보안(Security) 기능을 활성화하려면 필요한 환경변수 또는 설정을 모두 고르시오.

a. ROS_SECURITY_ENABLE=true
b. ROS_SECURITY_STRATEGY=Enforce
c. ROS_ENCRYPTION_LEVEL=HIGH
d. ROS_SECURITY_KEYSTORE=/path/to/keystore
e. ROS_SECURITY_ALLOW_UNSECURED=true

19. 다음 중 ROS2 에서 Component(Container 기반 실행)와 관련된 설명으로 올바른 것은?

a. Component 는 항상 독립된 프로세스에서 실행된다.
b. Component 는 colcon build 대신 make 로 빌드해야 한다.
c. Component 는 .so 파일로 빌드되어 컨테이너에 동적으로 로딩된다.
d. Component 는 ros2 run 으로만 실행 가능하며, launch 는 사용할 수 없다.
e. Component 는 QoS 설정을 포함할 수 없다.

20. 다음 Python 코드는 ROS2 퍼블리셔 생성 시 사용할 QoS 프로파일을 설정하는 부분이다. 빈칸 (A), (B), (C), (D)에 들어갈 코드를 올바르게 모두 고르시오.

```
from rclpy.qos import QoSProfile, (A), QoSDurabilityPolicy, (B)

qos = QoSProfile(
    reliability=(C),
    durability=QoSDurabilityPolicy.TRANSIENT_LOCAL,
    history=(D),
    depth=10
)
```

a. (A) → QoSReliabilityPolicy
b. (A) → QoSHistoryPolicy
c. (B) → QoSHistoryPolicy
d. (B) → QoSReliabilityPolicy
e. (C) → QoSReliabilityPolicy.RELIABLE
f. (C) → QoSReliabilityPolicy.KEEP_LAST
g. (D) → QoSHistoryPolicy.KEEP_LAST
h. (D) → QoSHistoryPolicy.RELIABLE
i. (D) → KEEP_LAST