

## 2024년도 전국기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅
<p>※ 다음 사항을 유의하여 채점하시오.</p> <ol style="list-style-type: none"> <li>1) AWS의 리전은 ap-northeast-2을 사용합니다.</li> <li>2) 웹페이지 접근은 크롬이나 파이어폭스를 이용합니다.</li> <li>3) 웹페이지에서 언어에 따라 문구가 다르게 보일 수 있습니다.</li> <li>4) Shell에서의 명령어의 출력은 버전에 따라 조금 다를 수 있습니다.</li> <li>5) 문제지와 채점지에 있는 &lt;&gt; 는 변수입니다. 해당 부분을 변경해 입력합니다.</li> <li>6) 채점은 문항 순서대로 진행해야 합니다.</li> <li>7) 삭제 채점은 되돌릴 수 없으므로 유의하여 진행합니다.</li> <li>8) 이의신청까지 완료 이후 선수가 생성한 클라우드 리소스를 삭제합니다.</li> <li>9) 부분 점수가 있는 문항은 채점 항목에 부분 점수가 적혀져 있습니다.</li> <li>10) 부분 점수가 따로 없는 문항은 전체 다 맞아야 점수로 인정됩니다.</li> <li>11) 채점 진행 전 환경 셋업을 위해 다음 사항을 확인해야 합니다. <ul style="list-style-type: none"> <li>- Bastion에 SSH로 접근할 수 있는지 확인합니다.</li> <li>- Bastion 접근 할 때 포트 2220으로 접속했는지 확인합니다.</li> <li>- Bastion에서 AWS CLI v2, curl, jq가 설치되어 있는지 확인합니다.</li> <li>- Bastion에서 IAM Role이 매핑되어 AWS CLI로 AWS의 모든 리소스에 접근 가능한지 확인합니다.</li> <li>- <code>aws sts get-caller-identity</code> 명령을 통해 선수의 계정이 아닌 다른 계정에 접근하고 있는지 확인합니다. 만약, 다른 계정이라면 부정행위를 의심할 수 있습니다.</li> </ul> </li> <li>12) 채점 전 채점환경 구성을 위해 ~/.aws/config 에 아래 내용이 추가되도록 합니다. <pre>[default] region = ap-northeast-2 output = json</pre> </li> <li>13) 채점 시에는 별도로 제공한 채점 스크립트(mark.sh)를 실행하여 채점할 수 있습니다. 다만, 선수가 직접 입력을 원할 경우 채점기준표에 명시된 명령어 그대로 입력하여 채점할 수 있습니다.</li> </ol>		

## 2. 채점기준표

1) 주요항목별 배점			직 종 명		클라우드컴퓨팅			
과제 번호	일련 번호	주요항목	배점	채점방법		채점시기		비고
				독립	합의	경기 진행중	경기 종료후	
제2과제	1	Code Commit	0.5		○		○	
	2	Code Build	0.5		○		○	
	3	Code Deploy	0.5		○		○	
	4	Code Pipeline	1		○		○	
	5	Automation	1.25		○		○	
합 계			3.75					

## 2) 채점방법 및 기준

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
2과제	1	Code Commit	1	Code Commit Configuration	0.25
			2	Source Code Location	0.25
	2	Code Build	1	Code Build Configuration	0.5
	3	Code Deploy	1	Code Deploy Configuration	0.5
	4	Code Pipeline	1	Pipeline Configuration	0.5
			2	Source, Build, Deploy Configuration	0.5
	5	Automation	1	V1 to V2	1.25
	총점				3.75

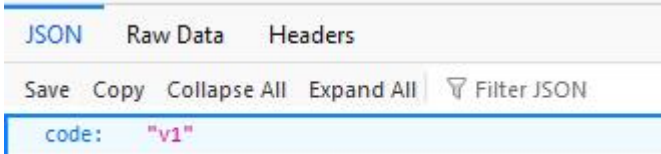
### 3) 채점내용

	채점 항목	
1-1	1-1A (명령어 입력)	aws codecommit list-repositories --query "repositories[].repositoryName" ₩ ; aws codecommit list-branches --repository-name wsi-repo
	1-1-A (예상 출력) <u>정확히</u> <u>일치</u> <u>순서 중요</u>	[ "wsi-repo" ] { "branches": [ "main" ] }
2-1	2-1A (명령어 입력)	aws codebuild list-projects ₩ ; aws ecr describe-repositories --repository-names wsi-ecr --query "repositories[].repositoryName"
	2-1-A (예상 출력) <u>정확히</u> <u>일치</u> <u>순서 중요</u>	{ "projects": [ "wsi-build" ] } [ "wsi-ecr" ]

3-1	3-1A (명령어 입력)	aws deploy list-applications ₩ ; aws deploy list-deployment-groups --application-name wsi-app ₩ ; aws deploy get-deployment-group --application-name wsi-app --deployment-group-name wsi-dg --query "deploymentGroupInfo.deploymentStyle.deploymentType"
	3-1-A (예상 출력) <u>정확히</u> <u>일치</u> <u>순서 중요</u>	{ "applications": [ "wsi-app" ] } { "deploymentGroups": [ "wsi-dg" ] } } "IN_PLACE"

4-1	4-1A (명령어 입력)	aws codepipeline get-pipeline --name wsi-pipeline --query "pipeline.name"
	4-1-A (예상 출력) <u>정확히 일치</u> <u>순서 중요</u>	"wsi-pipeline"

4-2	4-2-A (명령어 입력)	aws codepipeline get-pipeline --name wsi-pipeline --query "pipeline.stages[].name"
	4-2-A (예상 출력) <u>정확히</u> <u>일치</u> <u>순서 중요</u>	[ "Source", "Build", "Deploy" ]

	5-1-A (명령어 입력)	<p>wsi-server의 ip주소로 웹브라우저에 &lt;ip&gt;/v1/app을 호출합니다.</p> <p>codecommit 페이지에 이동하고 src/app.py의 7번째 줄에 v1을 v2 변경 합니다.</p>
5-1	<p>5-1-A (예상 출력) <u>정확히</u> <u>일치</u> <u>순서 중요</u></p>	<p>1.) &lt;ip&gt;/v1/app</p>  <p>2.) 7번째 줄에 v1에서 v2로 변경</p> <pre> 5 @app.route('/v1/app', methods=['GET']) 6 def get_app(): 7     return jsonify({"code": "v2"}) </pre> <p>3.) 변경 후 1분뒤 파이프라인이 완료되면 다시 &lt;ip&gt;/v1/app로 접속하면 v1을 v2로 변경되는 것을 확인 할 수 있습니다.</p> 