**Western Technical College**

# 10664100 Basic Robotic Programming

## Course Outcome Summary

### Course Information

| | |
|---|---|
| **Description** | In this course, learners are introduced to programming techniques for the Yaskawa DX200 robots. The learner examines teach pendant programming including I/O, routines, decision making, multiple axis of positional operation, and robot communication. Upon completion of the course, learners will be able to operate and program the Yaskawa DX200. |
| **Career Cluster** | Manufacturing |
| **Instructional Level** | Associate Degree Courses |
| **Total Credits** | 2 |
| **Total Hours** | 54 |

#### Textbooks

*Yaskawa DX200 Basic Programming with Material Handling Kit*. Publisher: Yaskawa Academy. Required.

### Success Abilities

1. Cultivate Passion: Increase Self-Awareness
2. Live Responsibly: Develop Resilience
3. Live Responsibly: Foster Accountability
4. Refine Professionalism: Act Ethically
5. Refine Professionalism: Improve Critical Thinking

### Program Outcomes

1. Perform work safely.
2. Communicate technical information.

### Course Competencies

**1. Identify components of the robotic pendant hardware.**

**Assessment Strategies**

1.1. Written Objective Test
1.2. Skill Demonstration

**Criteria**

*You will know you are successful when*

1.1. You locate the hardware on a multiple axis robotic controller pendant.
1.2. You describe the function of the hardware on a multiple axis robotic controller pendant.
1.3. You locate and manipulate basic controls on a multiple axis robotic controller pendant.

**Learning Objectives**

1.a. Identify the location of keys on the pendant.
1.b. Explore the function of keys on the pendant.
1.c. Differentiate between the pendant hardware and software.

## 2. Jog the robot using available coordinate systems.

**Assessment Strategies**

2.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

2.1. You differentiate between six-axis and Cartesian coordinate systems.
2.2. You jog the robot using individual six-axis coordinates.
2.3. You jog the robot using a Cartesian coordinate system.

**Learning Objectives**

2.a. Investigate Cartesian and six-axis coordinate systems.
2.b. Investigate custom user frame coordinate systems.
2.c. Investigate tool coordinate systems.
2.d. Display the robot motors' positions in pulse counts or position data.
2.e. Use the axis keys to jog the robot at different manual speeds.
2.f. Display the robot's tool center point position.

## 3. Explore various recovery procedures.

**Assessment Strategies**

3.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

3.1. You reset an error message.
3.2. You reset a minor alarm.
3.3. You display alarm history and details.
3.4. You perform overrun and internal shock sensor recovery.
3.5. You perform external shock sensor recovery.
3.6. You confirm position for the specified home point.

**Learning Objectives**

3.a. Cancel an error message.
3.b. Reset an alarm.
3.c. Access alarm history.
3.d. Perform shock sensor recovery.

## 4. Execute various project/task functions for robot motion.

**Assessment Strategies**

4.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

4.1. You create, display and select a project/task.
4.2. You create a new project/task, program steps, and verify playback with forward/reverse and interlock test start.
4.3. You operate the robot in teach, manual and automatic modes.
4.4. You delete, undelete, copy, and rename project/tasks.

**Learning Objectives**

4.a. Create a new program.
4.b. Display the active program.
4.c. Select a program.
4.d. Create a new program that contains six-axis and linear motions.
4.e. Program with the last step of program in same position as the first.
4.f. Perform path confirmation.
4.g. Play the new program in various cycle settings.
4.h. Display program list.
4.i. Display the program header.
4.j. Examine common program functions: copy, delete, undelete, rename.

## 5. Examine edit and motion instructions.

**Assessment Strategies**

5.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

5.1. You demonstrate line editing within a project/task and change motion types for a step within a job.
5.2. You explore the impact of using fine/point-to-point tag values in a motion step.
5.3. You explore different methods of changing speed tag data within a project/task.

**Learning Objectives**

5.a. Insert a step.
5.b. Modify a step's position.
5.c. Delete a step.
5.d. Change a step's motion type.
5.e. Explore common commands: copy, cut, paste, and undo.
5.f. Add tags to change the default motion instructions.
5.g. Examine methods to change speed tags.
5.h. Display the cycle time.
5.i. Edit job speeds.

## 6. Access a jogging plane for desired coordinates.

**Assessment Strategies**

6.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

6.1. You define a custom user frame.
6.2. You name a custom user frame.
6.3. You access custom user jogging coordinates.
6.4. You jog the robot in the custom user frame.
6.5. You edit a custom user frame.

**Learning Objectives**

6.a. Define a custom jogging plane.
6.b. Name a custom jogging plane.
6.c. Access jogging methods and select the desired plane.
6.d. Navigate to/from the various categories on the non-motion instruction menu.

## 7. Examine Input and Output instructions.

**Assessment Strategies**

7.1. Written Objective Test

7.2. Skill Demonstration

**Criteria**

*You will know you are successful when*

7.1. You access the programming language I/O list.
7.2. You monitor universal I/O.
7.3. You execute program instructions.

**Learning Objectives**

7.a. Simulate an Input.
7.b. Force an individual output.
7.c. Determine the binary value of 8-bits.
7.d. View the output as the tool center point passes through the interference zone.
7.e. Examine instructions needed to program outputs, waits, and inputs.
7.f. Use the stored byte-value for a designated purpose.
7.g. Program a wait instruction referencing the output for the interference zone.

## 8. Examine operating instructions.

**Assessment Strategies**

8.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

8.1. You program arithmetic instructions in a project/task.
8.2. You program increment and decrement instructions in a job.
8.3. You program set and clear instructions in a project/task.

**Learning Objectives**

8.a. Monitor the corresponding variable addresses.
8.b. Program a set instruction.
8.c. Program an increment instruction.
8.d. Program a decrement instruction.
8.e. Program a clear instruction.

## 9. Examine control instructions.

**Assessment Strategies**

9.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

9.1. You program call and return instructions within a project/job and monitor the parent and child programs.
9.2. You program structured text instructions within a project/job.
9.3. You program jump and jump label instructions within a project/task both with and without the use of conditional if instructions.

**Learning Objectives**

9.a. Explore and program various subroutine and nested jobs.
9.b. Monitor for the active program when using subroutines.
9.c. Explore ways to program conditional and unconditional subroutines.
9.d. Explore methods of calling or jumping to programs.
9.e. Examine timer, pause, and comment instructions.
9.f. Program using conditional statements.

## 10. Examine position variables.

**Assessment Strategies**

10.1. Skill Demonstration

**Criteria**

*You will know you are successful when*

10.1. You record position variables in six-axis and Cartesian coordinates .
10.2. You display the current position variable and are able to forward to that point.
10.3. You set and calculate an offset value in a position variable.
10.4. You program shift instructions within position variables using increment or decrement instructions.

**Learning Objectives**

10.a. Display variables.
10.b. Search for a specific variable.
10.c. Edit the value in a variable.
10.d. Format position variables to different types.
10.e. Set values into position variable elements.
10.f. Forward to position variables.
10.g. Set or calculate an offset value in a position variable.