This course explains about basic functions of GX Works3 for those who will use GX Works3 for the first time. Operation methods of GX Works3 are introduced as the learner configures an example programmable controller system through this course. In this course, programs are configured using the programming language called Ladder.
Introduction Purpose of the Course

This course explains about basic functions of GX Works3 for those who will use GX Works3 for the first time. Operation methods of GX Works3 are introduced as the learner configures an example programmable controller system through this course. In this course, programs are configured using the programming language called Ladder.

This course requires the basic knowledge of programmable controllers and the MELSEC Series programmable controllers.

The following courses are prerequisites prior to taking this course:

- FA Equipment for Beginners (PLCs)
- MELSEC iQ-R Series Basic
Introduction

Course structure

The contents of this course are as follows. It is recommended that you start from Chapter 1.

Chapter 1 - GX Works3 overview
Learn the fundamentals of GX Works3

Chapter 2 - System design
Learn about designing a programmable controller system

Chapter 3 - Program editing
Learn about creating control programs

Chapter 4 - Operation check
Learn about operation checks on created programs

Chapter 5 - Maintenance
Learn about maintenance after the system is in operation

Final Test
Pass grade: 60% or higher
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to the next page</td>
<td>Go to the next page.</td>
</tr>
<tr>
<td>Back to the previous page</td>
<td>Back to the previous page.</td>
</tr>
<tr>
<td>Move to the desired page</td>
<td>&quot;Table of Contents&quot; will be displayed, enabling you to navigate to the desired page.</td>
</tr>
<tr>
<td>Exit the learning</td>
<td>Exit the learning.</td>
</tr>
</tbody>
</table>
Safety precautions

When you learn based on using actual products, please carefully read the safety precautions in the corresponding manuals.

Precautions in this course

The displayed screens of the software version that you use may differ from those in this course. This course uses the following software version:

- GX Works3 Version 1.007H
Chapter 1  GX Works3 overview

GX Works3 is the programming and maintenance software specifically designed for the MELSEC iQ-R Series control system. GX Works3 consists of various different components that help to simplify project creation and maintenance tasks.
1.1 Scenes for GX Works3

The figure below shows typical lifecycle of a programmable controller system. GX Works3 can be used in all the scenes as shown below.

This course will introduce GX Works3 features in this order. In this course, programs are configured using the programming language called Ladder.
1.2 Summary

In this chapter, you have learned:

- GX Works3 overview

Important points to consider:

| GX Works3 overview | GX Works3 consists of various different components that help to simplify project creation and maintenance tasks. |
Chapter 2 System design

This chapter explains about designing a programmable controller system.

2.1 Programmable controller system example
2.2 Components for the example system
2.3 Main features of GX Works3
2.4 Creating a project
2.5 Module configuration according to the system
2.6 Setting module operations
2.7 Giving names to devices
2.8 Saving the created content
2.9 Summary
2.1 Programmable controller system example
2.2 Components for the example system

The example label inspection system requires the following components.

- **Conveyor**
- **Output module**
- **Input module**
- **Captured image**
- **Proximity sensor 1**
- **Vision sensor**
- **Proximity sensor 2**
- **Reject mechanism**

### Programmers and Controllers

<table>
<thead>
<tr>
<th>Programmable controller</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU module</td>
<td>R04CPU</td>
</tr>
<tr>
<td>Base unit</td>
<td>R35B</td>
</tr>
<tr>
<td>Power supply module</td>
<td>R61P</td>
</tr>
<tr>
<td>Input module</td>
<td>RX40C7</td>
</tr>
<tr>
<td>Output module</td>
<td>RY10R2</td>
</tr>
</tbody>
</table>

### External equipment

<table>
<thead>
<tr>
<th>External equipment</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proximity sensors 1, 2</td>
<td>Detects bottle positions.</td>
</tr>
<tr>
<td>Vision sensor</td>
<td>Checks if a label is correctly pasted on a bottle.</td>
</tr>
<tr>
<td>Reject mechanism</td>
<td>Pushes out a bottle with a defective label.</td>
</tr>
<tr>
<td>Conveyor</td>
<td>Conveys bottles to the sensors and the reject mechanism.</td>
</tr>
</tbody>
</table>
2.3 Main features of GX Works3

Screen layout of GX Works3 should be understood before designing a system. Place the mouse cursor over a window or an area to learn about its functions.

Element Selection window
Elements available for programming are listed. Elements can be dragged and dropped onto the Work window. A required element can be searched, and often used elements can be added to Favorites.
2.3 Main features of GX Works3
2.4 Creating a project

Click to proceed to the next section.
To view again, click on the “Replay” button.
2.5 Module configuration according to the system

The next section explains how to automatically read an existing system configuration directly from the hardware.

Click to proceed to the next section.

To view again, click on the “Replay” button.
2.5.1 Reading the actual module configuration

Click \( \text{Play} \) to proceed to the next section.
To view again, click on the "Replay" button.
The next section shows how to fix the module configuration.

Click to proceed to the next section.

To view again, click on the “Replay” button.

Replay
2.5.3 Fixing the module configuration

Click 🎁 to proceed to the next section.
To view again, click on the “Replay” button.
Setting module operations

Click to proceed to the next section.

To view again, click on the “Replay” button.
2.7 Giving names to devices

Device names, which are processed by programmable controllers, can have labels for easier understanding. A label name can be information such as device usage or connected device. By showing such information as labels, program contents become more easily understandable.

I/O devices corresponding to external equipment are assigned with the following labels. Details are given in Chapter 3.

<table>
<thead>
<tr>
<th>External equipment</th>
<th>Device</th>
<th>Input or output</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proximity sensor 1</td>
<td>X0</td>
<td>Input</td>
<td>ProximitySensor_1</td>
</tr>
<tr>
<td>Vision sensor</td>
<td>X1</td>
<td>Input</td>
<td>VisionSensorResult</td>
</tr>
<tr>
<td>Proximity sensor 2</td>
<td>X2</td>
<td>Input</td>
<td>ProximitySensor_2</td>
</tr>
<tr>
<td>Reject mechanism</td>
<td>Y10</td>
<td>Output</td>
<td>PusherStart</td>
</tr>
<tr>
<td>Conveyor</td>
<td>Y11</td>
<td>Output</td>
<td>ConveyorStart</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Label</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ProximitySensor1_Count</td>
<td></td>
</tr>
<tr>
<td>ProximitySensor2_Count</td>
<td></td>
</tr>
<tr>
<td>DefectiveLabelCount</td>
<td></td>
</tr>
<tr>
<td>LastDefectiveLabelCount</td>
<td></td>
</tr>
<tr>
<td>PushCompleteTime</td>
<td></td>
</tr>
<tr>
<td>PushCompleteTimer</td>
<td></td>
</tr>
<tr>
<td>PushTrigger</td>
<td></td>
</tr>
</tbody>
</table>
This section explains different types of labels before proceeding to the explanation of label registration procedure. The main two types are Global Label and Local Label, and they differ by their applicable ranges.

**Global Label**

Global Label can be used for different programs within a project.

<table>
<thead>
<tr>
<th>Label Name</th>
<th>Data Type</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Double Word [Unsigned]/Bit String [32-bit]</td>
<td>VAR_GLOBAL</td>
</tr>
<tr>
<td>Year</td>
<td>Double Word [Unsigned]/Bit String [32-bit]</td>
<td>VAR_GLOBAL</td>
</tr>
</tbody>
</table>

The next section shows how to actually register a Global Label. Click to proceed to the next. To view again, click on the "Replay" button.
Global Label registration

To execute the Reservation to Register/Release for the system label, reflection to the system label database is required. Please execute 'Reflect to System Label Database'.

It is unnecessary to change reference side project when assigned device is changed in system label Ver.2.

* Only IQ-R series/GOT 2000 series is available for system label Ver.2.
* To execute Online Program Change, execute Online Program Change and save.

Click to proceed to the next section.

To view again, click on the "Replay" button.
### 2.7.3 Local Label registration

![Image of software interface with local label registration]

- **Click** to proceed to the next section.
- To view again, click on the “Replay” button.

---

<table>
<thead>
<tr>
<th>Label Name</th>
<th>Data Type</th>
<th>Class</th>
<th>Initial Value</th>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProximitySensor1_Count</td>
<td>Double Word [Unsigned]/Bit String</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
<tr>
<td>DefectiveLabelCount</td>
<td>Double Word [Unsigned]/Bit String</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
<tr>
<td>LastDefectiveLabelCount</td>
<td>Double Word [Unsigned]/Bit String</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
<tr>
<td>ProximitySensor2_Count</td>
<td>Double Word [Unsigned]/Bit String</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
<tr>
<td>PushCompleteToTime</td>
<td>Word [Unsigned]/Bit String [16-bit]</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
<tr>
<td>PushCompleteTimer</td>
<td>Timer</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
<tr>
<td>PushTrigger</td>
<td>Bit</td>
<td>VAR</td>
<td></td>
<td>VAR</td>
</tr>
</tbody>
</table>

---

**Navigation:**
- Project
- Edit
- Find/Replace
- View
- Online
- Debug
- Diagnostics
- Tool
- Window
- Help
2.7.4 Module Label overview

System configuration is completed.
Click to proceed to the next section.
To view again, click on the "Replay" button.
2.8 Saving the created content

Click \( \text{Play} \) to proceed to the next section.

To view again, click on the “Replay” button.
2.9 Summary

In this chapter, you have learned:

- Programmable controller system example
- Components for the example system
- Main features of GX Works3
- Creating a project
- Module configuration according to the system
- Setting module operations
- Giving names to devices
- Saving the created content

Important points to consider:

<table>
<thead>
<tr>
<th>Module configuration</th>
<th>Module configuration of GX Works3 is a graphical diagram showing a physical module configuration. Basic parameters can also be set from this diagram.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>Easily recognizable names can be assigned as labels to make a program more understandable.</td>
</tr>
<tr>
<td>Global Label</td>
<td>Global Labels can be used in multiple programs within a project.</td>
</tr>
<tr>
<td>Local Label</td>
<td>Local Labels can be used in a program.</td>
</tr>
<tr>
<td>Module Label</td>
<td>Module Label is a label already assigned to an I/O address or a buffer memory address of a specific module.</td>
</tr>
</tbody>
</table>
Chapter 3  Program editing

This chapter explains about creating control programs.

3.1 Programming languages and their characteristics
3.2 System specifications
3.3 Program contents
3.4 Editing a program
3.5 Using grouped instructions
3.6 Making a program understandable
3.7 Creating comments in multiple languages
3.8 Checking the program for mistakes
3.9 Converting a program to the executable format
3.10 Summary
3.1 Programming languages and their characteristics

Operations of a programmable controller must be written out as a control program. GX Works3 supports the following programming languages. Various different programming languages can be used within the same project.

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Features</th>
</tr>
</thead>
</table>
| Ladder               | • In ladder programming, contacts and coils are used to create a program resembling an electrical circuit.  
                        • Instruction processes are easy to follow even for a user with little experience. |
| FBD (Function Block Diagram) | • In FBD, a program consists of function blocks.  
                                • Program contents are easily seen and are easily reproduced. |
| ST (Structured Text)  | • ST program is described using texts.  
                        • ST may be familiar for programmers who have experience in C programming. |
| SFC (Sequential Function Chart) | * Coming soon  
                                   • Conditions and processes are described in a flow chart.  
                                   • Program steps are easy to follow. |

This course will use Ladders in creating the example inspection system program.
Before proceeding to program editing, please confirm the specification of the example system.

### I/O devices

<table>
<thead>
<tr>
<th>External equipment</th>
<th>Input or output</th>
<th>Global Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proximity sensor 1</td>
<td>Input</td>
<td>ProximitySensor_1</td>
</tr>
<tr>
<td>Vision sensor</td>
<td>Input</td>
<td>VisionSensorResult</td>
</tr>
<tr>
<td>Proximity sensor 2</td>
<td>Input</td>
<td>ProximitySensor_2</td>
</tr>
<tr>
<td>Reject mechanism</td>
<td>Output</td>
<td>PusherStart</td>
</tr>
<tr>
<td>Conveyor</td>
<td>Output</td>
<td>ConveyorStart</td>
</tr>
</tbody>
</table>

### Internal devices

<table>
<thead>
<tr>
<th>Label name (Local Label)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProximitySensor1_Count</td>
</tr>
<tr>
<td>ProximitySensor2_Count</td>
</tr>
<tr>
<td>DefectiveLabelCount</td>
</tr>
<tr>
<td>LastDefectiveLabelCount</td>
</tr>
<tr>
<td>PushCompleteTime</td>
</tr>
<tr>
<td>PushCompleteTimer</td>
</tr>
<tr>
<td>PushTrigger</td>
</tr>
</tbody>
</table>
3.3 Program contents

This section explains about the program required for the example inspection system. Here is the example inspection system and the control program linked with the system operation.

Normal operation

Please click the button below to start the animation.

Play

Click ⏯️ to proceed to the next.
To view again, click on the upper left "Play" button.
3.5 Using grouped instructions

In a program, often-used instructions can be grouped together as a function block (FB). FB can simplify a lengthy program and shorten the programming time.

An FB can be created by the user or a selection of FBs can be obtained from your local Mitsubishi Electric representative. GX Works3 also has pre-made FBs called Module FB. The Module FB is specific to a module and contains a set of instructions that are typically used.

Click **Replay** to proceed to the next section.

To view again, click on the “Replay” button.
3.5.1 Creating a program containing Module FB

The bottle label inspection system uses a relay output module to control the reject mechanism. Although the relay type output module can handle large load current as it uses mechanical contacts, (which can have a limited service life) the internal relay contact would require servicing. To allow for this maintenance, a program that notifies the service life is required and can be done easily using a Module FB.

Please click the button below to start the animation.

Play

Warning lamp turns ON.

 Relay ON count 250,001

Click to proceed to the next.
To view again, click on the upper left "Play" button.
3.5.2 Placing a Module FB

Click to proceed to the next section.
To view again, click on the “Replay” button.
3.6 Making a program understandable

- **(1) Initial settings**
  - SM402 ON once after CPU RUN
  - MOV P K10 PushCompleteTime

- **(2) Defective label processing**
  - ProximitySensor
  - Detects that a bottle reached the vision sensor

- **(3) Reject arm processing**

Click ☀️ to proceed to the next.
To view again, click on the “Replay” button.
### 3.7 Creating comments in multiple languages

![Image of software interface showing comments in multiple languages](image)

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Japanese/日本語</th>
<th>English (Display Target)</th>
<th>Chinese/中文</th>
</tr>
</thead>
<tbody>
<tr>
<td>M0</td>
<td>ルート設定値への到達ON</td>
<td>On when relay limit reached</td>
<td>到達路器設定値時ON</td>
</tr>
<tr>
<td>M1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M24</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M27</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M28</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M34</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M35</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M36</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M37</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M39</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M40</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Click 🎁 to proceed to the next.
To view again, click on the “Replay” button.
3.8 Checking the program for mistakes

Click \( \bullet \) to proceed to the next.

To view again, click on the "Replay" button.
3.9 Converting a program to the executable format

Click ▶️ to proceed to the next section.
To view again, click on the “Replay” button.
3.10 Summary

In this chapter, you have learned:

- Programming languages and their characteristics
- System specifications
- Program contents
- Editing a program
- Using grouped instructions
- Making a program understandable
- Creating comments in multiple languages
- Checking the program for mistakes
- Converting a program to an executable format

Important points to consider:

<table>
<thead>
<tr>
<th>FB</th>
<th>Various instructions that are used multiple times are grouped together in a function block (FB). FB can simplify a lengthy program and shorten the overall programming time. An FB can be created by the user, or utilize one of the FB pre-installed in GX Works3.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module FB</td>
<td>Module FB is an FB specific to a module and contains a set of instructions typically used for the module.</td>
</tr>
<tr>
<td>Comment</td>
<td>Make a program more understandable to the programmer and the others. Reduces chances of programming mistakes. Entered in multiple languages.</td>
</tr>
<tr>
<td>Program conversion</td>
<td>Required to be converted to a format executable by the programmable controller CPU module.</td>
</tr>
</tbody>
</table>
Chapter 4  Operation check

This chapter explains how to check the operation of created programs.

4.1 Confirming the example inspection system
4.2 Debugging using the simulation function
4.3 Debugging on the actual system
4.4 Preparing for the system operation
4.5 Summary
Confirming the example inspection system

The program to inspect labels and the program to detect the relay limit life are shown here.

Please click the button below to start the animation.

Click to proceed to the next.
To view again, click on the upper left "Play" button.
4.2 Debugging using the simulation function

Click \( \Rightarrow \) to proceed to the next section.
To view again, click on the “Replay” button.
4.2 Debugging using the simulation function

Click ▶️ to proceed to the next section.
To view again, click on the “Replay” button.
4.3 Debugging on the actual system

Resetting the CPU module

Executing control programs

Executing control programs

P RUN LED turns on, and the control program is executed.

Click ⏩ to proceed to the next section.
To view again, click on the “Replay” button.
4.3 Debugging on the actual system

Click to proceed to the next section.
To view again, click on the “Replay” button.
4.4 Preparing for the system operation

Click to proceed to the next.
To view again, click on the “Replay” button.
4.5 Summary

In this chapter, you have learned:

- Confirming the bottling label inspection system
- Debugging using the simulation function
- Testing on the actual system
- Preparing for the system operation

Important points to consider:

<table>
<thead>
<tr>
<th>Simulation feature</th>
<th>The simulation function checks the program operation without physical modules.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring function</td>
<td>The program being executed can be monitored by using the monitoring function.</td>
</tr>
</tbody>
</table>
Chapter 5 Maintenance

This chapter explains how to maintain a system by using GX Works3.

5.1 Checking for any abnormalities
5.2 Investigating the error cause
5.3 Maintenance at overseas locations
5.4 Course summary
5.1 Checking for any abnormalities

Preliminary diagnosis can be performed by looking at CPU module LED lamps. Flashing "BAT LED" indicates an error related to the battery.

The next section explains more how to do this. Click to proceed to the next section. To view again, click on the “Replay” button.
5.2 Investigating the error cause

Click \( \square \) to proceed to the next section.
To view again, click on the "Replay" button.
5.3 Maintenance at overseas locations

Click to proceed to the next section.

To view again, click on the “Replay” button.
5.4 Course summary

The program for the bottling label inspection system has been successfully completed, and the system is confirmed to be operating normally. This brings us to the end of this e-Learning course.

GX Works3 is the essential software in configuring control programs for MELSEC programmable controller systems.
5.5 **Summary**

In this chapter, you have learned:

- Checking for any abnormalities
- Investigating the error cause
- Maintenance at overseas locations
- Course summary

Important points to consider:

<table>
<thead>
<tr>
<th>Comments in multiple languages</th>
<th>When using the created program at overseas sites, the comment language can be switched according to the language spoken by the local maintenance engineer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagnosis function</td>
<td>When the system operates abnormally, connecting a computer where GX Works3 is installed to the programmable controller will start up automatic diagnosis.</td>
</tr>
</tbody>
</table>
Now that you have completed all of the lessons of the Engineering Software MELSOFT GX Works3 (Ladder) course, you are ready to take the final test. If you are unclear on any of the topics covered, please take this opportunity to review those topics.

There are a total of 7 questions (7 items) in this Final Test. You can take the final test as many times as you like.

**How to score the test**

After selecting the answer, make sure to click the Answer button. Your answer will be lost if you proceed without clicking the Answer button. (Regarded as unanswered question.)

**Score results**

The number of correct answers, the number of questions, the percentage of correct answers, and the pass/fail result will appear on the score page.

<table>
<thead>
<tr>
<th>Correct Answers</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Questions</td>
<td>9</td>
</tr>
<tr>
<td>Percentage</td>
<td>22%</td>
</tr>
</tbody>
</table>

To pass the test, you have to answer **60%** of the questions correct.

- Click the **Proceed** button to exit the test.
- Click the **Review** button to review the test. (Correct answer check)
- Click the **Retry** button to retake the test again.
Overview of GX Works3

Please select the correct description about GX Works3. (Multiple answers)

- [ ] Software must be switched according to the usage such as system design, startup, and maintenance.
- [ ] GX Works3 can be used in various stages of the product development lifecycle such as system design and maintenance.
- [ ] Different programming languages cannot be used within the same project.
- [ ] The simulation function enables the program operation to be checked without requiring physical modules.
- [ ] On a program, comments can be added in different languages, and the displayed language is switchable.
Label types

Please select the correct description about labels. (Multiple answers)

- A Global Label can be used in multiple programs.
- A Local Label can be used in multiple programs.
- Easily recognizable names can be assigned as "labels" to make a program more understandable.
- Labels Increase the processing speed of programs.
Overview of FB

Please select the correct description about FB. (Multiple answers)

- Often-used instructions can be grouped as an FB.
- Custom FBs cannot be created.
- FB will simplify a large program.
- Programming time is reduced by grouping often-used Instructions as an FB.

- FB stands for Function Bank.
Overview of Module FB and Module Label
Please select the correct description about Module FB and Module Label. (Multiple answers)

- Module FB contains a set of instructions typically used for a specific module.
- Module Labels can be used without considering I/O and buffer memory addresses.
Overview of comments

Please select the correct description about comments. (Multiple answers)

- By having comments, the program becomes more understandable.
- Comments make a program more understandable and also reduce any mistakes.
- If the program is used overseas, comments can be added in the local language to make the program content understandable in the local language.

- Comments are automatically translated into the selected language.
- Comments are used to show the version of the program.

Back
Comment types

Which type of comments is added to a ladder rung? Please select an answer.

- Device/label comment
- Statement
- Note

Back
Automatic diagnosis
When a system error occurs, the diagnosis feature of GX Works3 is automatically started just by connecting to a computer. Please select the correct connection method between the computer and the CPU module.

- Ethernet connection
- USB connection
You have completed the Final Test. Your results are as follows:
To end the Final Test, proceed to the next page.

Correct answers: 7
Total questions: 7
Percentage: 100%

Congratulations. You passed the test.
You have completed the Engineering Software MELSOFT GX Works3 (Ladder) course.

Thank you for taking this course.

We hope you enjoyed the lessons and the information you acquired in this course will be useful in the future.

You can review the course as many times as you want.

Review  Close