Programming Concepts
(Part B)

ENGR 10
Introduction to Engineering
Clicker Question 1

What are the values of A and B at end of this program?

A = 10 ; 
B = 12 ;
if ((A + B) < 22)
{  B = A ; 
   A = B;   }
else
{  A = B;  
   B = A + B; }  
A)  A=12, B=22  
B)  A=12, B=10  
C)  A= 10, B=10  
D)  A=12, B=24  
E)  A=10, B=22
Functions

What is a function?

Function is just a group of code given a name. By grouping and naming it we can reuse it anywhere in the application without having to rewrite the whole block of code again.

Why use a function?

It saves time and reduces the overall code size of your program. But more importantly, if there is a problem you only have to fix it in one place.

The following example shows how to write a function and how to call that function.

MAX_Three → find the maximum of 3 numbers
How to Define a Function in easyCPro

Right click on User Function and select Add New Function

Define the Name and Return Type

```c
int MAX_Three (void)
{
    int max = A;
    if (B > A)
    {
    } // End if
    max = B;
    if (C > A && C > B)
    {
    } // End if
    max = C;
    return max;
}
```
New controller (Cortex)

List of all functions and the Main program
Sequential execution of main program which calls the defined function

A, B, C, Z1, and Z2 are defined as global variables

```c
void main ( void )
{
    A = 10;
    B = -2;
    C = 7;
    Z1 = MAX_Three();
    A = B + C;
    B = C + 2;
    C = 32;
    Z2 = MAX_Three();
}
```

```c
int MAX_Three ( void )
{
    Variables
    BEGIN
    max = A;
    IF
    {
        if ( B > A )
        {
            max = B;
        }
    } IF
    {
        if ( C > A && C > B )
        {
            max = C;
        }
    }
    return
    return max;
    EIND
    } END
```
Introduction to Local Variable

- The MAX_Three function finds the greatest of the three numbers A, B and C.

- These variables were defined as global variables.

- Before this function is called, your program must have copied the values into these three variables.

- This operation can be done more efficiently by using local variables.
Program using Function with passing variables

void main ( void )
{
    A = 10;
    B = -2;
    C = 7;
    Z1 = MAX_Three ( A, B, C );
    P = A + B;
    Q = B + A - C;
    R = A - C;
    Z2 = MAX_Three ( P, Q, R );
}

int MAX_Three ( int x, int y, int z )
{
    max = x;
    
    if ( y > x )
    {
        max = y;
    }
    
    if ( z > x && z > y )
    {
        max = z;
    }
    
    return max ;
}

x, y, and z are defined as Local variables

A, B, and C are NOT defined as global variables
What is a Local variable?

• The value of a local variable is accessible only to the function within which it is defined.
• The main function (main program) passes the values into the function and these values are copied into the local variables of the function.
• The function doesn’t care what are the names of the variables defined in the main function.
• The function does what it should do and then returns the result to the main function.
• The values of these local variables are lost after the execution of the function is completed.
Global Variable Concept

A, B, C, and Z are defined as global variables

Main Function

Defined Function

Task of the function

A
B
C

Z
Local Variable Concept

Main Function

Defined Function

$x, y, \text{ and } z$ are defined as Local variables

Task of the function

A → $x$ → max → Z
B → $y$ → max → Z
C → $z$ → max → Z
A local variable is known only to the function in which it is defined:

A. True
B. False
Mechanical Components – Motors (EasyC)

1. **Motor**, continuous rotation of the shaft

2. **Servo**, shaft rotates through a specified angle, used with both old and new controllers

Make sure the clutch module is always used. It prevents damage to the motor in case of large torque (resistance to motion).
Mechanical Components – Motor (Cortex)

The new robot kit includes three 2-wire motors that can be converted to a 3-wire motor using the controller 29. Ports 1 and 10 on the new controller are designated for 2-wire motors only.
Motor

• Robot movement made possible by the use of motors.
• The robot uses two motors, one controlling each side of the robot, steering the robot.
• This combination is enough to perform any of the basic movements.
• The motors can be rotated either clockwise or counter-clockwise, controlling the overall motion of the robot.
This chart describes the combination of the motor movement required to perform each maneuver.
Continuous motor

Select Motor, drag and drop on the line between Begin and End

Choose motor number, as connected to the Controller

Max. speed clockwise

Max. speed counterclockwise

Custom speed

Hsu/Youssefi
## Motor Speed & Direction

<table>
<thead>
<tr>
<th>Value</th>
<th>Motor Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>255</td>
<td>Clockwise high speed</td>
</tr>
<tr>
<td>254</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>Clockwise low speed</td>
</tr>
<tr>
<td>127</td>
<td>Stop</td>
</tr>
<tr>
<td>126</td>
<td>Counter Clockwise low speed</td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Counter Clockwise high speed</td>
</tr>
</tbody>
</table>
Continuous motor (Cortex)
How to move a motor?

• Use the following statement to move a motor

```c
SetMotor ( 2, 60 );
```

This number indicates that the motor is connected to motor port 2 of the total available 8 ports (10 ports on Cortex)

This number sets the direction and speed of the motor
How to connect the motor to the Controller?

• On the VEX controller you have several ports available so that you can connect not only motors but also other devices and sensors to it.

PORT:

• A port is a specific place on the board where we can connect other external devices, so that the two devices in connection can exchange information or data.
Controller (EasyC)

Motor ports

Sensor ports
The section labeled “Analog/Digital” ports on the controller

Ports 5 – 8 are for bumper sensors and limit switches

Ports 1 – 4 are for light sensors

Ports 9 and 10 are preprogrammed for stops when bumper sensor is activated

These ports are taken by the blue connector, connected to the IRB mounted on the robot.
Controller (Cortex)

Motor and Servo ports

8 analog and 12 digital ports. Digital ports are used for Limit, Bumper and Ultrasonic sensors.
Side view of the Vex Controller

Battery Ports (Square and round connectors)
Two 3-Wire motors to operate the claw (3-pin)

Two continuous motors to drive the robot (2-pin)
Program to move the robot forward

• Now let us write a program to make the robot move forward for 2 seconds and then stop.

Assume:

Left Motor – connected to motor port 3
Right Motor – connected to motor port 2
Program to make the motor move forward for 2 seconds and then stop

```c
void main ( void )
{

    SetMotor ( 3 , 255 ) ; // left motor connected to port 3 is set to move clockwise

    SetMotor ( 2 , 0 ) ; // right motor connected to port 2 is set to move counter-clockwise

    Wait ( 2000 ) ; // This statement pauses the execution of the program for 2000 ms

    SetMotor ( 3 , 127 ) ; // left motor is stopped

    SetMotor ( 2 , 127 ) ; // right motor is stopped

    Wait ( 2000 )
}
```
Clicker Question 3

What value of $x$ will cause the motor for Cortex (new) controller to be stopped: $\text{SetMotor}(2, x)$?

A. 64
B. 127
C. 0
D. 255
E. -127
Servos (EasyC)

- Servo units are similar to motors except that they control position rather than speed.
- Servo units have limited range of motion (about 120°)
- A value between 0 and 255 sets the direction of the servo.
  - 0  – Extreme clockwise direction
  - 127  – No movement
  - 255  – Extreme counter-clockwise direction
Graphical representation of the direction of rotation of the Servo (EasyC)
Servo

• Similar to the motor the following statement is used to set the servo to turn to a direction.

\[
\text{SetServo ( 5 , 64 )];}
\]

Indicates the motor port number to which the servo is connected.

This value sets the direction and the rotation of the servo.
SetServo (5, 64) - EasyC

Motor shaft rotates 30° clockwise and then stops
SetServo (5, 191) - EasyC

Motor shaft rotates 30° counterclockwise and then stops.
Servos (Cortex)

• Servos are similar to motors except that they control position rather than speed.
• Servo units have limited range of motion (about 120°)
• A value between -127 and 127 sets the direction of the servo.
  127 – Extreme clockwise direction
  0 – No movement
  -127 – Extreme counter-clockwise direction
SetServo (5, 64) - Cortex

Motor shaft rotates 30° clockwise and then stops.
Program showing how to move the servo continuously

```c
void main ( void )
{
    while ( 1 )
    {
        SetServo ( 5 , 64 ); // servo connected to port 5 is turned about 30 deg clockwise
        Wait ( 1000 ); // pause the program for 1 sec
        SetServo ( 5 , 191 ); // servo connected to port 5 is turned about 30 deg counter-clokwise
        Wait ( 1000 ); // pause the program for 1 sec
    }
}
```
Output of the above program
Mechanical Components - Sensors

- Limit Switch Sensor (Digital)
- Bumper Switch Sensor (Digital)
- Ultrasonic Sensor (Digital)
Introduction to Logic Signals

- A logic signal can take on either 0v or 5v
  - 0v represents “Low state” or “Logic 0”
  - 5v represents “High state” or “Logic 1”

A logic signal can also be called a “digital signal”
Bumper Switch

- A Bumper Switch is a digital sensor.
- It can distinguish between two states: pressed or not pressed (On or Off)
- When the switch is not pressed, the robot interprets this value as a 1 (high state).
- When the switch is pressed the robot interprets the value as a 0 (low state).
• Use of the following statement in the program to read the bumper switch.

```c
Bumper = GetDigitalInput( 2 );
```

A variable into which the status of the bumper switch will be stored. This indicates the port number to which the bumper switch is connected.
Limit Switch

• A limit switch behaves just like a bumper switch
• When switch is pressed: digital signal = 0
• When switch is open (not pressed): digital signal = 1
• Program the same way as bumper
stores the state of the bumper switch into the variable “bumper”.

checks if there is a contact with an obstacle (switch will be pressed if there is a contact)

motor moves forward as long as there is no obstacle (bumper ==1)

if the switch is pressed (bumper==0), these statements make the robot halt for 1 second

Then the robot turns left for 2 seconds (green arrow is ccw)
Digital Output

• The “Digital output” function can be used to control the state of a digital output.

• By using this function you can send digital signals (either 0 or 1) to the devices which can accept digital signals.

• For example, by using this function you can turn off or on an LED.

  1 => 5v => light is on
  0 => 0v => light is off
Digital Output

• This statement is used in the program to control the devices i.e., send digital signals.

```
SetDigitalOutput ( 11 , 1 );
```

Indicates the port number (here port 11) to which the device to be controlled is connected

The digital value to be sent (either 0 or 1)
Drag and drop the Digital Output

Set the output signal (range), 0 or 1
Program to make an LED blink continuously

```c
void main ( void )
{
    while ( 1 )
    {
        SetDigitalOutput [ 11 , 1 ]; // makes the LED connected to port 6 ON by sending digital signal 1
        Wait ( 1000 ); // keeps the LED ON for 1 sec
        SetDigitalOutput [ 11 , 0 ]; // makes the LED connected to port 6 OFF by sending digital signal 0
        Wait ( 1000 ); // keeps the LED OFF for 1 sec
    }
}
```
What voltage value represents “High State”? 

A. 0 V 
B. 10 V 
C. 5 V 
D. 255 
E. 127
Analog Signals

• For the VEX controller, an analog signal can take any value between 0v and 5v.

→ 0v is represented by a value 0
→ 5v is represented by a value 1023 \( (2^{10}) \)

• All other voltage levels between 0v and 5v are represented by the numbers between 0 and 1023.
Analog Input

• An analog input function block can be used to receive analog signals (any value between 0v & 5v) from analog sensors.

• For our VEX controller, the range 0v-5v is represented by the range of numbers 0-1023.

• But in general the range used to represent 0v-5v depends on the controller.
Ultrasonic Sensor

• The Ultrasonic Sensor is a digital sensor.
• The Ultrasonic Sensor uses high-frequency sound waves to detect objects.
• It emits a sound wave, measures how long it takes the sound wave to bounce back, and then calculates how far away the object is.
• The measurement is translated into a numeric value from 2 - 100.
• Lower signal value corresponds to a closer detected object and vice versa.
Ultrasonic Sensor Connections

VEX Controller

Ultrasonic Sensor

Digital Output port

Interrupt port

Input

Output
Ultrasonic Sensor

INPUT
Sends ultrasonic wave, connected to Digital

OUTPUT
Receives the echo, connected to INTERRUPT
Ultrasonic Sensor

The following statement starts an ultrasonic sensor i.e., the sensor starts sending and recording the sound signals.

```
StartUltrasonic ( 1, 8 ) ;
```

Sensor connected to output port 8 through which it receives digital control signals from the controller.

Sensor connected to interrupt port 1 of the total available 6 ports.
Ultrasonic Sensor

The following statement is used to read a value from the sensor:

\[ \text{Range} = \text{GetUltrasonic} (1, 8); \]

Variable into which the translated value is stored:

Interrupt port #1 to which the sensor is connected:

The sensor is connected to output port #8 on the controller:

First define variable Range:

![Diagram of Ultrasonic Sensor configuration](image-url)
Configuring port 8 from a digital input to a digital output port

Double click on the Config. icon

Click on the circle to change the direction of the arrow.

Taken by the blue connector to the IRB
Program to stop the forward moving robot when a nearing object is detected

```c
void main ( void )
{
  SetMotor ( 3 , 255 );
  SetMotor ( 2 , 0 );
  StartUltrasonic ( 1 , 8 );
  while ( 1 )
  {
    Range = GetUltrasonic ( 1 , 8 );
    if ( Range <= 10 )
    {
      SetMotor ( 3 , 127 );
      SetMotor ( 2 , 127 );
    }
  }
}
```

Setting the robot to move forward

Start the ultrasonic detector connected to interrupt port # 8 and output port # 1

Store the translated signal into the variable Range

Checking if the obstacle is near or far

Making the robot to stop if an obstacle is nearby
Change port 6 from input to output by clicking on the circle.

Digital ports taken by the IRB.
void main ( void )
{
  SetMotor ( 1 , 127 ) ;
  SetMotor ( 10 , -127 ) ;
  StartUltrasonic ( 5 , 6 ) ;
  while ( 1 )
  {
    Range = GetUltrasonic ( 5 , 6 ) ;
    if ( Range<=10 )
    {
      SetMotor ( 1 , 0 ) ;
      SetMotor ( 10 , 0 ) ;
    }
  }
}

Input to Ultrasonic sensor connected to output port 6 on controller
Output from Ultrasonic sensor connected to input port 5 on controller
Light Sensor

• A light sensor is an example of an analog sensor.
• A light sensor reads the intensity of light of an area, translates the reading into a single numeric value, and sends that value to the robot.
• The range of values used by this sensor is 0-1023
• The higher the value, the darker will be the area and vice versa.
Light sensor

• The following statement is used to read the analog signal from a light sensor.

    light = GetAnalogInput ( 1 ) ;

A variable named “light” into which the reading (0-1023) from the light sensor is stored

Indicates the port number (here port 1) to which the light sensor is connected
Program to make the robot move towards the moving light source

```c
void main ( void )
{
    while ( 1 )
    {
        light_left = GetAnalogInput ( 7 ); // analog input from light sensor at port 7 is stored into light_left
        light_right = GetAnalogInput ( 8 ); // analog input from light sensor at port 8 is stored into light_right
        if ( light_left > light_right )
        {
            SetMotor ( 3 , 97 );
            SetMotor ( 2 , 97 );
        }
        else if ( light_left < light_right )
        {
            SetMotor ( 3 , 157 );
            SetMotor ( 2 , 157 );
        }
        else
        {
            SetMotor ( 3 , 177 );
            SetMotor ( 2 , 77 );
        }
    }
}
```

If light intensity is more on the left side, the robot turns toward the left side.

If light intensity is more on right side, the robot turns toward the right side.

If light intensity is the same on both sides, the robot moves forward.
Optical Shaft Encoder

- The optical shaft encoder is a digital sensor. It is used to measure rotational movement.
- As the disc rotates, an infrared light sensor is used to count the number of slots passed.
- A count of 90 makes one revolution.
Optical Shaft Encoder

• The following statement is used to start the encoder counting.

StartEncoder ( 2 ) ;

This number indicates that the encoder is connected to interrupt port # 2 on the VEX controller.
Optical Shaft Encoder

• The following statement is used to get the present count value from the encoder.

```
Count = GetEncoder (2);
```

A variable named “count” into which the present count value will be stored

Interrupt port #2 to which the Encoder is connected
PresetEncoder ( 2, 25 );
This statement is used to make the encoder start counting from a given number (here 25) instead of from zero.

Example
Write a program to make the robot move for 1 meter and then stop.
Assume 5 turns of the wheel = 1 meter
Program to make the robot move forward for 1 meter

Presetting the encoder to 20. Count value starts from 20.

```
void main ( void )
{
    StartEncoder ( 2 );
    PresetEncoder ( 2 , 20 );
    SetMotor ( 3 , 177 );
    SetMotor ( 2 , 77 );
    while ( 1 )
    {
        count = GetEncoder ( 1 );
        if ( count == 470 )
        {
            SetMotor ( 3 , 127 );
            SetMotor ( 2 , 127 );
            StopEncoder ( 2 );
        }
    }
}
```

if count reaches 470 the robot stops.
470 = 450 (count for 5 turns) + 20 (initial value)
E10 robot, bumper switches, it wonders

E10 robot – ultrasonic sensor

E10 new robot