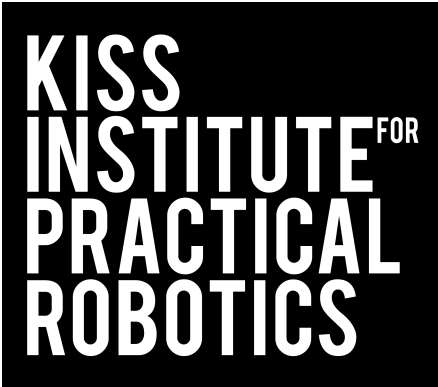
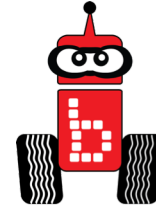


Introduction to Programming Languages

KIPR Module 3

Table of Contents



Introduction to Programming Languages

Activity 3.1 - Understanding Programming Languages

- Programming Languages
- Why Not Use an Interpreter
- Terms to Understand

KIPR Software Suite

Looking at a Program

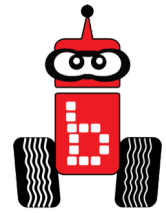
Activity 3.2 - Identify Parts of a Program

Activity 3.3 - Rules and Conventions

Activity 3.4 - How do word processing skills help you write computer code?

Activity 3.5 - Keyboarding Skills

Introduction to Programming Languages



Goals

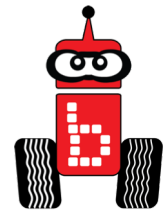
- To help students understand that languages have rules/conventions you must follow
- Understand how programs such as Microsoft Word or other applications help you follow the rules/conventions of a program
- Understand that the KISS IDE (Integrated Development Environment) is like a word processing program that helps them follow the rules/conventions while they write their source code
- New Vocabulary word is Debug; meaning checking your program for errors and fixing them, much like when Microsoft Word underlines misspelled words or grammatical errors
- To help students understand the basic components of a **C** program

Understanding Programming Languages

Activity 3.1

1. Have the students read and discuss the following slides:
 - Programming Languages
 - Why Not Use an Interpreter
 - Terms to Understand
2. Use a strategy such as chant it, sing it, rap it to share the information.

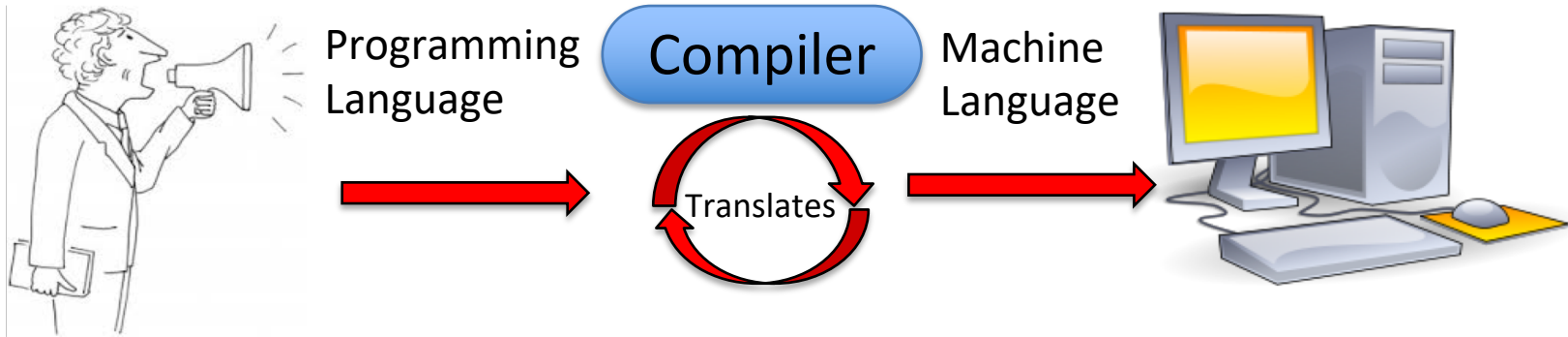
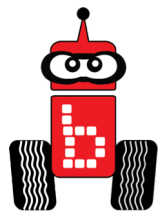
Programming Languages



Computers only understand machine language (stream of bytes), which they can then read and execute (run).

Humans on the other hand, don't do well with machine language.

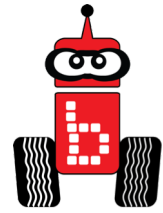
Why Not Use an Interpreter?



Humans have created languages with funny names like; C, C++, JAVA and Python, that allow them to write “source code” which they can understand and edit.

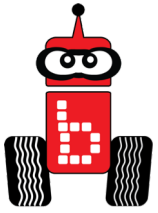
This source code is then compiled (translated) into machine language that the computer can understand and execute (run).

Terms to Understand

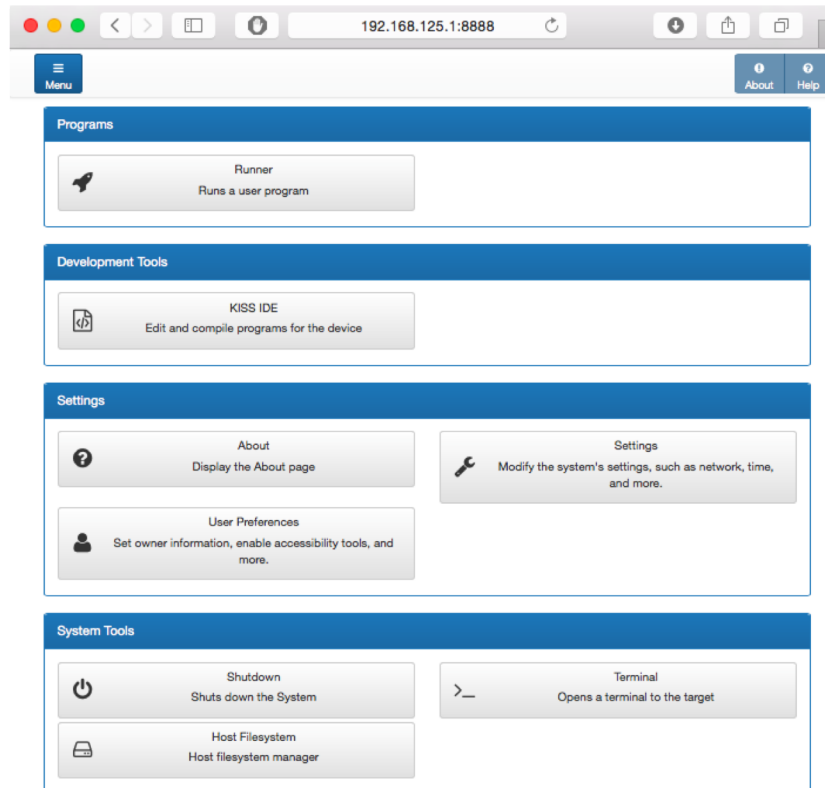


- o **Machine Language** -- what the computers understand- Bytes
- o **Executes** -- in terms of a computer running or carrying out the instructions
- o **Source Code** -- name for code written in programming language
- o **Compiled** -- translated from a programming language to a machine language
- o **Programming Language** -- Language humans understand that can be turned into machine language
- o **C, C++, Java, Python** -- names of programming languages
- o **IDE** -- Integrated Development Environment

KIPR Software Suite



We will be using the **KIPR Software Suite as our IDE** to develop source code with the **C** programming language. KIPR Software Suite is a software application that makes it easy for you to write and edit your source code, debug it (look for and correct mistakes) and compile (translate) it.



Looking at a Program

1. Use a strategy such as the Jigsaw strategy to learn about C program. Use the following slides:

int main

Curly Braces

Code

return

Semi-colons

Colors

Looking at a Program

`int main`

return type *name* *argument list*



```
int main()  
{  
    return 0;  
}
```

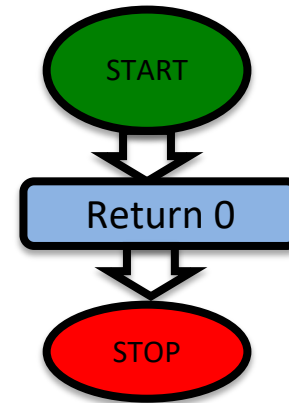
- The `int main` is the main program, `int` stands for integer (whole number)
- The curly braces bracket the programming statements
- Programs should always return something, in this case it is a `0`

Looking at a Program

Curly Braces

Start → `int main()`
Stop → `{`
`}`

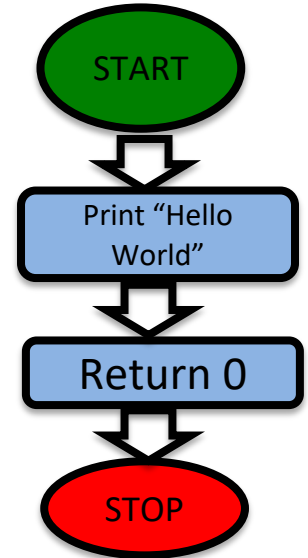
```
int main()  
{  
    return 0;  
}
```



The curly braces organize everything you want the program to do (execute) when the computer comes to the last curly brace it will end the main program. The actions you want the robot to do will be between the curly braces.

Looking at a Program Code

```
int main()  
{  
    printf("Hello, World!\n");  
    return 0;  
}
```

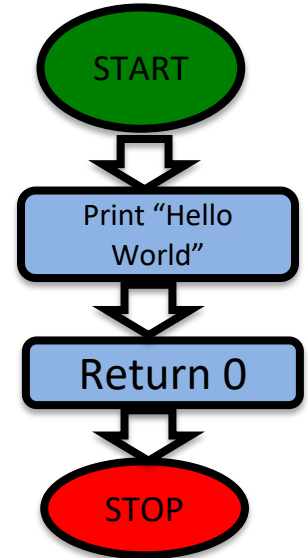


This is the code and specifies the things (functions) you want the program to execute.

Looking at a Program

return

```
int main()  
{  
    printf("Hello, World!\n");  
    return 0;  
}
```

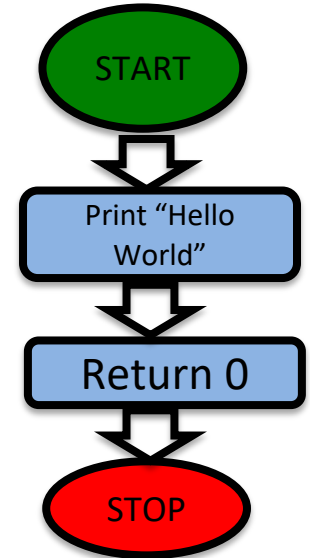


Notice the program returns a value even though it is 0.

Looking at a Program Semicolons

```
int main()  
{  
    printf("Hello, World!\n");  
    return 0;  
}
```

Diagram illustrating the role of semicolons in the C code. Red circles highlight the semicolons at the end of the `printf` and `return 0` statements. Red arrows point from these semicolons to the explanatory text below.

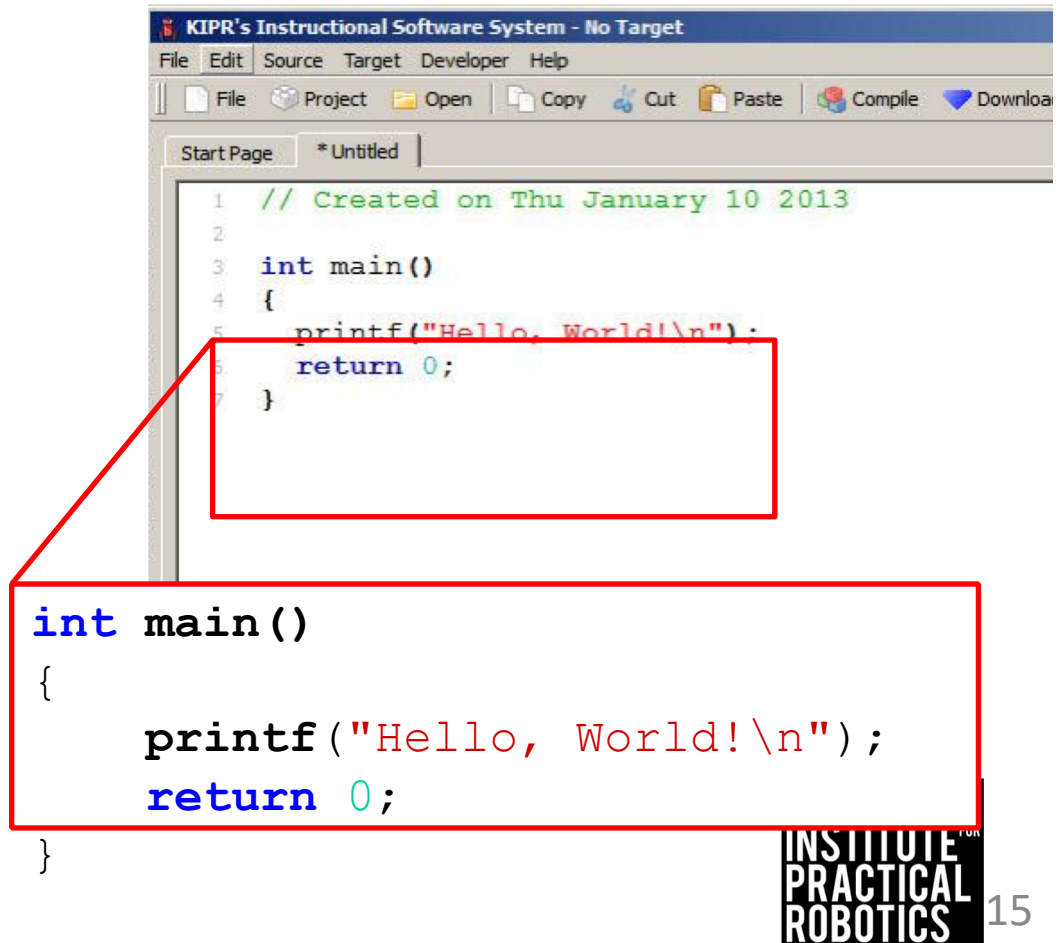


When the program is executing, the semicolon terminates the code and says go to the next line. Without it, the code will not compile (be translated so the computer can understand it).

Looking at a Program Colors

The KISS IDE highlights parts of a program to make it easier to read. By default, the KISS IDE colors your code and adds line numbers

- Comments appear in **green**
- Key words appear in **bold blue**
- Text strings appear in **red**
- Numbers appear in **aqua**



KISS's Instructional Software System - No Target

File Edit Source Target Developer Help

File Project Open Copy Cut Paste Compile Download

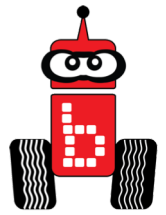
Start Page *Untitled

```
1 // Created on Thu January 10 2013
2
3 int main()
4 {
5     printf("Hello, World!\n");
6     return 0;
7 }
```

int main()
{
 printf("Hello, World!\n");
 return 0;
}

INSTITUTE
PRACTICAL
ROBOTICS

Identify Parts of a Program



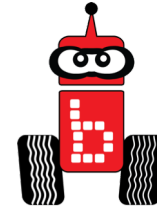
Activity 3.2

1. Discuss the “Looking at a Program” slides.
2. Label and explain what the following parts do from the Sample Program on the next slide.
 - Curly braces
 - Semi-colons
 - Main function
 - Return
 - Source code (programming statements)
 - Color of the text in your code

Sample Program

```
int main()  
{  
    printf("Hello, World!\n");  
  
    motor(3,100);  
    motor(0,100);  
    msleep(3800);    //forward  
  
    motor(3,100);  
    motor(0,25);  
    msleep(1000);    //turn right  
  
    return 0;  
}
```

Rules and Conventions



Activity 3.3

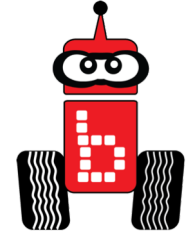
These are a few of the rules that are always done in a language.

- When writing a sentence you always start with a capital letter.
- Complete sentences should end with a period.
- Spaces are used to create paragraphs, which are used to separate ideas.
- The order of words make a difference
 - *The ran horse slowly; The horse ran slowly.*
- Math has conventions as well
 - Order of operations $(1+2) \times 4$

What other conventions or rules did you think of?

Discuss the different rules/conventions and use the POPCORN SHARE Strategy to share.

How do word processing skills help you write computer code?



Activity 3.4

Materials: computer and/or print out of a keyboard

1. Have students use the Think-Pair-Share Strategy on how word processing programs, such as Microsoft Word or Google Docs, work as you use them.
 - How do you input information?
 - What keys on the keyboard are good to know?
 - What tricks help you with shortcuts?
1. Discuss how writing computer programming is like typing in a word processing program.

Keyboard Skills

Activity 3.5

Students will learn how to use important keyboard skills that will help them with programming.

Materials: keyboard print out or computer/laptop keyboard

1. Look at the keyboard or use a laptop/computer and identify the following keys:

- { } Curly braces keys
- () parenthesis keys
- return/enter key
- ; semicolon key
- delete/back
- arrow keys
- \ forward slash key
- <> greater than, less than keys
- = equal key
- number keys
- " quotation mark key

2. Practice holding down the shift button and the following keys at the **same time**:

- { } () "

3. You may want to teach older students how to copy and paste.

Keyboard



Assessments and Rubrics



Suggestions: *Understanding* or *Group Collaboration* rubrics