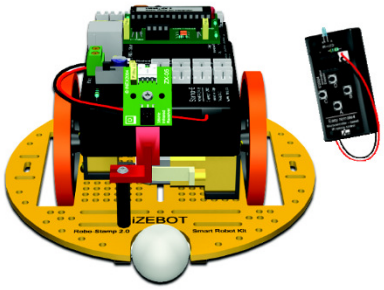
CMPS3111 Programming Project 2
DUE DATE: Thursday 31 October 2024
FORMAT: Digital Submission (zipped)
TOTAL POINTS: 112 (12% of overall course grade)

INSTRUCTIONS:

You will work with your team to complete the tasks for this assignment.

A template for the cover page is provided on the CMPS3111 Moodle site. This cover page includes a signature block. Team members are to sign if they have contributed to and are knowledgeable of the work done for the submission. **Team member names that appear on the signature block without a signature will not receive a grade. Grades will be awarded based on the level of contribution indicated by the "% of Total" column.**

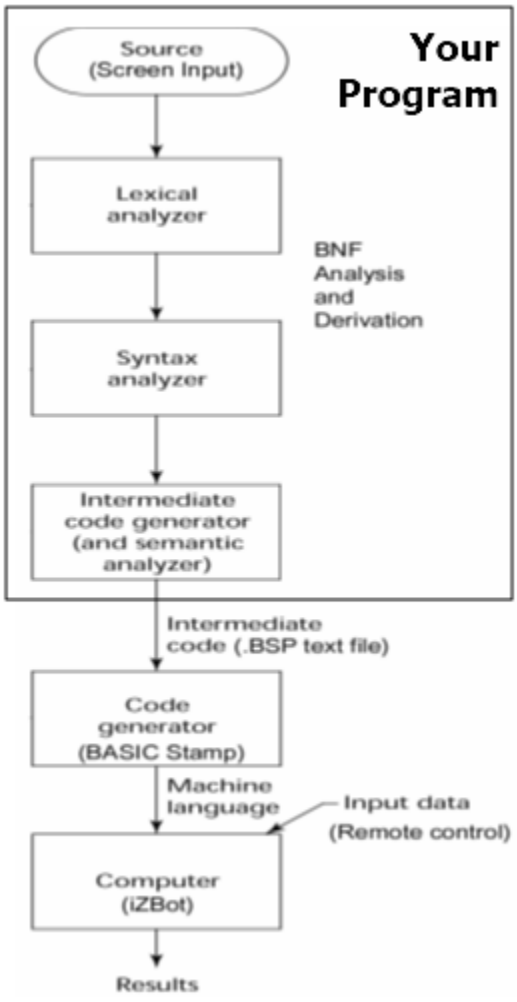
The Robot



The Robo-Stamp 2P uses the Basic Stamp 2P (i-Stamp2P24 Microcontroller Robotic Controller Board) to control how it works. The Basic Stamp 2P instructions are written in the PBASIC programming language using the Basic Stamp Editor. The Basic Stamp Editor uploads PBasic programs as machine code to the Robo-Stamp 2P via the computer's serial port. The Robo-Stamp 2P can be configured as the iZEBOT or the RoboTank.

For this exercise, we will use the Robo-Stamp 2P configured as the iZEBOT that can be controlled via infrared light using a remote control.

The Problem



As discussed in class, you will construct a program that will be a part of an iZEBOT meta-language pseudo-compiler process.

Your program will be responsible for the part of the process that:

- accepts an input sentence representing a potential program;
- recognizes through derivation whether the input is a valid sentence of the meta-language grammar;
- If the input is a valid sentence, it outputs intermediate code as a PBASIC program to the screen and in a text file with the extension BSP.

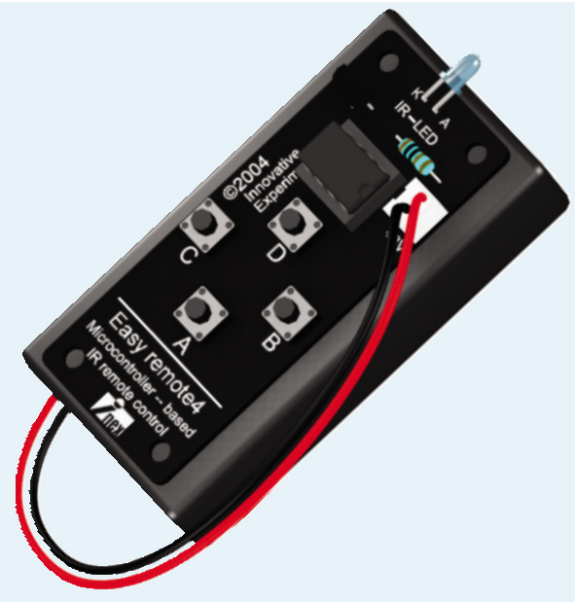
The pseudo compiler process is completed by uploading the machine code representation of your outputted PBASIC program to the iZEBOT via the BASIC Stamp Editor.

The program is then executed on the robot.

The iZEBOT meta-language will provide for the easy programming of four keys (A, B, C, D) of the iZEBOT remote control from 6 available movements:

1. Move forward, represented by the instruction: **DRIVE**
2. Move backwards, represented by the instruction **BACK**
3. Turn left, represented by the instruction: **LEFT**
4. Turn right, represented by the instruction: **RIGHT**
5. Spin left represented by the instruction: **SPINL**
6. Spin right represented by the instruction: **SPINR**

The meta-language will have the structure of:



wake
key ***x*= *y;***
⋮
[key ***x*= *y***;]
sleep

Where:

- 1 or more remote keys may be programmed
- ***x*** is the letter (a, b, c, or d) of the key being programmed
- ***y*** is the movement (DRIVE, BACK, LEFT, RIGHT, SPINL, or SPINR) assigned to the key being programmed

Examples:

wake
key a = DRIVE;
key b = BACK;
key c = LEFT;
key d = RIGHT;
sleep

All four keys are programmed.

wake
key a = SPINL;
key d = DRIVE;
sleep

Only keys a and d are programmed.

The Solution

A. Write a BNF grammar for the meta-language for the iZEBOT remote control that strictly adheres to the specification provided above.

B. The BNF grammar may be written using BNF or EBNF.

C. Write a computer program that:

(a) Upon starting it displays the BNF grammar for the meta-language for iZEBOT remote control
(b) Accepts an input string.
(c) If the user enters "END" when prompted for an input string, the program terminates.
(d) Conduct lexical/ syntax analysis on the input string to verify that is a sentence of the meta-language by attempting a LEFTMOST derivation based on the meta-language grammar on the input string. The derivation attempt is outputted to the screen.
    i. If the derivation is unsuccessful, it generates an appropriate error, the program pauses and prompts the user to press a key or click to continue, and then returns to (a).
    ii. If the derivation is successful, it indicates this and then proceeds to:
        1. pause and prompt the user to press a key or click to continue;
        2. draws the PARSE TREE for the derivation of the input string and displays the tree;
        3. pause and prompt the user to press a key or click to continue;
        4. generate the PBASIC (intermediate) program for the sentence accepted by inserting relevant header, footer and subroutine instructions from the PBASIC code provided or constructing the required body instructions;

5. then displays onscreen the program generated;
6. and saves the program generated to a **text** file named IZEBOT.BSP;
7. pause and prompt the user to press a key or click to continue;
8. Loops back to (a).

(e) The machine code for the BSP file will then be uploaded to the iZEBOT via the BASIC Stamp Editor.
(f) The correctness of the uploaded program will be verified by running the iZEBOT.

D. The generated PBASIC program will comprise a HEADER, a BODY, a FOOTER 1, a SUBROUTINE, and a FOOTER 2 code blocks (provided below). The HEADER and FOOTER blocks are static while the BODY and SUBROUTINE blocks are dynamic, both dependent on the key selected for programming and the movements assigned to the keys selected.

See the slides from the class discussion for an example of a program generated for a valid sentence.

The following is the necessary code and format for the respective blocks:

HEADER BLOCK code

```
'{$STAMP BS2p}
'{$PBASIC 2.5}
KEY         VAR     Byte
Main:       DO
            SERIN 3,2063,250,Timeout,[KEY]
```

BODY BLOCK - Button assignment code (1 or more lines depending on how many keys are programmed)

```
IF KEY = "x" OR KEY = "y" THEN GOSUB routine
```

Where ***x*** is A, B, C, or D;
y is a, b, c, or d;
routine is Forward, Backward, TurnLeft, TurnRight, SpinLeft, SpinRight, or Motor_OFF

FOOTER 1 Code

```
            LOOP
Timeout:    GOSUB Motor_OFF
            GOTO Main

'+++++ Movement Procedure ++++++++++++++++++++++++++++++
```

SUBROUTINE BLOCK code

```
Forward:   HIGH  13 : LOW 12 : HIGH 15 : LOW 14 : RETURN
Backward:  HIGH  12 : LOW 13 : HIGH 14 : LOW 15 : RETURN
TurnLeft:  HIGH  13 : LOW 12 : LOW  15 : LOW 14 : RETURN
TurnRight: LOW   13 : LOW 12 : HIGH 15 : LOW 14 : RETURN
SpinLeft:  HIGH  13 : LOW 12 : HIGH 14 : LOW 15 : RETURN
SpinRight: HIGH  12 : LOW 13 : HIGH 15 : LOW 14 : RETURN
```

FOOTER 2 code

```
Motor_OFF: LOW   13 : LOW 12 : LOW  15 : LOW 14 : RETURN
'++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Syntactic structure of generated PBASIC program

Header Code
    Body Code *(content depends on the button and the movements selected)*
Footer Code 1
Subroutine Code *(include only relevant subroutines for the movements selected)*
Footer Code 2

The Program [42 points]

A. The program should comprise:

   (a) A MAIN that displays the BNF grammar for the meta-language and then prompts for an input string. If the input string is "END" the program terminates, else MAIN should call:

      i. A subprogram that executes a LEFTMOST derivation of the input string showing each sentential form and the final generated sentence. If the derivation cannot be completed successfully (i.e. the string is not recognized by the grammar), the subprogram should generate an appropriate error message. The subprogram then prompts the user to press a key or click to continue and returns to (a).

      ii. If the derivation is successful, a subprogram draws the PARSE TREE for the derivation of the input string, displays the tree, and then prompts the user to press a key or click to continue and returns to (a).

      iii. A subprogram that is called after subprogram (ii) if the derivation is successful which:

         1. generates the PBASIC program code by applying the relevant semantics for the input string,
         2. displays onscreen the generated code,
         3. and saves the generated code in a **text** file named IZEBOT.BSP
         4. and then prompts the user to press a key or click to continue and returns to (a).

B. The program should be written in the designated languages:

- Julia
- Crystal
- Zig
- Ring

Program Presentation [32 points]:

A. In class your team will display (via PowerPoint) and explain:

   a. the BNF grammar for the meta-language for iZEBOT remote control
   b. the program code and its logic

   This part of your presentation should not be longer than five-(5) minutes.

   Your team will then demonstrate the compilation and execution of the program (the lecturer will provide input strings). Before the demo, the required compiler/interpreter must be installed on the computer used for the demonstration. This part of your presentation should not be longer than ten-(10) minutes.

B. Your entire presentation will not last longer than 15 minutes. **Practice prior to class since you will be graded on your timing. Also, come prepared since the clock starts when your team is called on to present.**

Document Submission [38 points]:

The report document should be assembled as a PDF document with the cover page containing the signature block in a professional format and will comprise three sections:

A. Descriptions and explanations for:

   a. the BNF grammar for the language with a brief description of what it represents
   b. a derivation example of a sentence of the language
   c. the brief description of the language used (flavour, version, compiler/interpreter, etc.)
   d. the program objectives, the process flow of the program, and a description of the program main and subprograms.

B. A concise but easily readable flow chart of the program. Be sure to check that you are using the correct symbols.

C. The program code with appropriate comments and remarks for improved readability (important for grading).

The programming project submission should be:

   a. Submitted BEFORE the START of class via email;
   b. Submitted as a ZIP file containing the Report PDF document, the PowerPoint presentation, and the program code.

For more detailed grading information, please review this document's Grading Criteria and Presentation Rubric.

Additional information for the programming project is provided in the document titled "Program 2: Supplemental Information" on the course Moodle page.

Groups

GROUP 1	**GROUP 2**	**GROUP 3**	**GROUP 4**
Julia	**Crystal**	**Zig**	**Ring**
Garbutt, Mickali	Aban, Kelsey	Coleman, Aiyesha	Vasquez, Addie
Garcia, Enrique	Hope, Christian	Forman, Daryn	Ramirez, Jerry
Kukul, Pedro	Garcia, Immanuel	Requena, D'Alesseo	Tillett, Victor
Pinelo, Aiden	Guerra, Alex	Rosado, Leonidez	Tzib, Enrisen
Shol, Julius		Ticas, Josue	Vasquez, Amilcar

GRADING CRITERIA
CMPS3111 Programming Project 2

1	Program	42.00
a.	Compilation Demonstration	2.00
b.	Invalid inputs w/ derivation	6.00
c.	Valid inputs w/ derivation	6.00
d.	Parse Tree / Intermediate file generation display	8.00
e.	Upload and robot code execution	10.00
f.	Program Evaluation	
	Program Interface	2.00
	On-screen instructions	2.00
	Program feedback	2.00
	Error handling	2.00
	Program Flow	2.00
2	Presentation	32.00
a.	Organization	4.00
b.	Visual Aids	4.00
c.	Mechanics	4.00
d.	Eye Contact	4.00
e.	Verbal Techniques	4.00
f.	Subject Knowledge	4.00
g.	Content	4.00
h.	Timing	4.00
3	Document Submission: Report	33.00
a.	Cover page	2.00
b.	BNF grammar explanation w/ derivation example	5.00
c.	Language (flavor, version, compiler/interpreter, etc.)	2.00
d.	Description of the program objectives	2.00
e.	Description of the program process flow	3.00
f.	Description of Main	5.00
g.	Description of Subprograms	5.00
h.	Flow Chart	3.00
i.	Flow Chart Symbols	2.00
j.	Program Code	
	Flow/Readability	2.00
	Comments/Remarks	2.00
k.	Late submission	
4	Document Submission: Programming Project	5.00
a.	Report in PDF format	2.00
b.	PowerPoint presentation	1.00
c.	Program code	1.00
d.	Zipped	1.00
e.	Late submission	
	Total Value/ Points	112.00
	VALUE/ GRADE	**12.00**

PRESENTATION RUBRIC

Presentation Rubric				
	1	2	3	4
Organization	Audience cannot understand presentation because there is no sequence of information.	Audience has difficulty following presentation because student jumps around.	Student presents information in logical sequence which audience can follow.	Student presents information in logical, interesting sequence which audience can follow.
Visual Aids	Student uses superfluous visual aids or no visual aids.	Student occasionally uses visual aids that rarely support the presentation.	Student's visual aids relate to the presentation.	Student's visual aids explain and reinforce the presentation.
Mechanics	Student's presentation has four or more spelling errors and/or grammatical errors.	Presentation has three misspellings and/or grammatical errors.	Presentation has no more than two misspellings and/or grammatical errors.	Presentation has no misspellings or grammatical errors.
Eye Contact	Student makes no eye contact and only reads from notes.	Student occasionally uses eye contact, but still reads mostly from notes.	Student maintains eye contact most of the time but frequently returns to notes.	Student maintains eye contact with audience, seldom returning to notes.
Verbal Techniques	Student mumbles, incorrectly pronounces terms, and speaks too quietly for audience in the back of class to hear.	Student's voice is low. Student incorrectly pronounces terms. Audience members have difficulty hearing presentation.	Student's voice is clear. Student pronounces most words correctly. Most audience members can hear presentation.	Student uses a clear voice and correct, precise pronunciation of terms so that all audience members can hear presentation.
Subject Knowledge	Student does not have grasp of information about subject.	Student is uncomfortable with information, provides only rudimentary information and fails to elaborate.	Student is at ease and provides most information with explanations and some elaboration.	Student demonstrates full knowledge (more than required) by providing information with explanations and elaboration.
Content	Overview of topic	Use of examples	Relevance of material presented	Correctness of material presented
Timing	<8 mins	8-10 minutes	10-12 or >15 minutes	12-15 minutes