CSI 402 – Program II

Administrative Information:

- **Deadline:** 11 PM, Monday, March 6, 2006.
- Two parts, but just one `makefile`.
- Two or more C source files for each part.
- README file (by 10 PM, Feb. 23, 2006):

  `~csi402/public/prog2/prog2.README`

Part (a): (Weightage: 30%)

- **Goal:** Empirical study of binary search trees.
- **Command line:**

  % `p2a` *infile* *outfile*

- The input file is a text file containing one string per line.
- Initially, the tree is empty.

- Create a binary search tree by inserting each string from the file into the tree.
- No rebalancing.
- Compute and print to the output file (also a text file) the following quantities.
  (a) No. of strings in the input file.
  (b) Height of the binary search tree.
  (c) No. of leaves in the tree.
  (d) Height of the left subtree of the root.
  (e) No. of strings in the left subtree of the root.
  (f) Height of the right subtree of the root.
  (g) No. of strings in the right subtree of the root.

Errors to be detected: See handout.

Part (b): (Weightage: 70%)

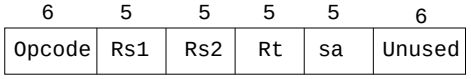
- **Goal:** An assembler for TMIPS ("Tiny MIPS").
- **Requirement:** Symbol table *must* be implemented as a binary search tree.

Brief Information about TMIPS:

- Memory size $= 2^{16} =$ 65,536 words. (Successive words have addresses 0, 1, 2, . . ., 65535.)
- Word size = 32 bits.
- Instruction length = 32 bits.
- 32 registers, each of size 32 bits. (Registers are denoted by $0, . . ., $31.)
- Supports only the `int` data type; 32-bit integers (2's complement form).
- Addressing modes: Direct, Base + Displacement and Immediate operands.

Instruction Formats:

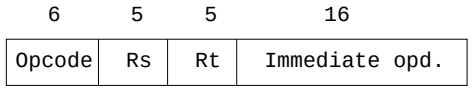
(a) R-Format:



Examples:

```
add     $3,$2,$1
move    $5,$3
```


(a) I-Format:

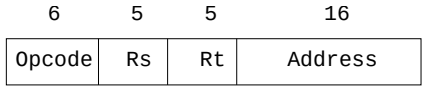


Note: Immediate operand is in 2's complement form with bit 15 as sign bit.

Examples:

```
addi    $3,$2,-7
swb     $5,-2($4)
```

(c) J-Format:



Note: Address treated as a 16-bit unsigned integer.

Examples:

```
j       next
jgt     $7,$9,loop
lwa     $5,val
```

TMIPS Assembly Language:

- Each line has at most 80 characters (including '\n').
- Blank lines allowed.
- Comments start with '#'.
- Fields of an instruction:
  - Label (optional); if present, terminated by ':'. (Conditions for valid labels given in the handout.)
  - Opcode.
  - Operands (optional).
  - Inline comment (optional – also begins with '#').
- List of machine opcodes: See handout.
- List of pseudo-opcodes (directives): `.text`, `.data`, `.resw` and `.word`.
- If both data and text segments are present, data segment comes after the text segment.

Example:

```
        .text
        lwa     $5,arr  #$5 has -7.
        swa     $5,x1   #Store -7 at x1.
        .data
x1:     .resw   10
arr:    .word   -7:15
```

Assembler Conventions: (See handout for details.)

- Starting address of the program is zero.
- Locations created using `.resw` must be initialized to zero.
- Unused bits in an instruction must be set to zero.

Errors to be detected: See handout.

Command line:

% `p2b` *infile*

- Input file has TMIPS assembly language program.
- Creates several output files.

- Object file.
- List file.
- Error file.

- Naming conventions for output files: To be discussed in class. (Details are in the handout.)

Object File Format:

- A text file.
- Each line has address and contents (both hexadecimal values) separated by one tab.

Note: Use `"%x"` format with `fprintf` to print an `int` or `short` value in hexadecimal.

List File Format:

- Also a text file.
- Contains each line of source file (including comments and blank lines) preceded by line number.
- Also contains the symbol table in *dictionary* order.

Error File Format:

- Also a text file.
- Each line has a line number followed by a brief error message.
- Line numbers must be in *increasing* order.
- List of multiply defined symbols and undefined symbols given after all the error messages.

Remarks:

- Assembler should not stop at the first error; it must check all the source lines for errors.
- If a line has two or more errors, report any one of the errors.
- Check for the (usual) command line errors.

Examples of assembly: To be presented in class.