Prof B's External Functions

Save the following external functions in a lib folder that is within your CS 3B directory.

barnett_12Feb20.a (https://canvas.ivc.edu/courses/44008/files/5481293/download?wrap=1)

- ascint32

  -
    ```
    @ Subroutine ascint32: This method converts a string of characters to an equivalent 4-byte (word) value.
    @                      The binary value of the converted ASCII string is returned in the R0 register.
    @                      If there is an invalid character in the numeric string which would invalidate the
    @                        conversion the Carry Flag is set to 1 and the R0 register returns 0.
    @                      If the numeric string contains a number that is too large to fit in a word,
    @                        the Over Flow is set to 1 and the R0 register returns 0.
    @
    @ R0: Must point to a null terminated string
    @ LR: Contains the return address

    @ Returned register contents:
    @      R0: Decimal result
    @ All registers are preserved except R0.
    ```
- hexToChar

  -
    ```
    @ Subroutine hexToChar: This method converts hex values to ascii and stores it into
    @                       the string passed to it in R1.
    @
    @ R0: Contains the hexvalue to be converted to ascii
    @ R1: Must point to address of a string large enough to hold the converted value.
    @ R2: Number of nibbles to be displayed (from right side of R0)
    @     Note: If R2=0 or R2>8, leading zeroes (on left) will not be displayed.
    @ LR: Contains the return address

    @ All registers are preserved.
    ```
- intasc32

  -
    ```
    @ Subroutine intasc32: This method converts a binary word to a printable string of ASCII
    @                       characters. You have to provide the address of a string that is
    @                       large enough to hold the converted value. Usually a string of 12
    @                       bytes is more than sufficient to allow for a sign as well as the
    @                       largest possible value a word could be.
    @
    @ R0: Contains the binary (signed) value to be converted to ascii
    @ R1: Must point to address of a string large enough to hold the converted value.
    @ LR: Contains the return address

    @ All registers are preserved except R0.
    ```
- putch

  -
    ```
    @ Subroutine putch: Provided a pointer to a character, putch will
    @      display the character to the terminal
    @ R0: Must point to a character byte
    @ LR: Must contain the return address
    @ All registers are preserved.
    ```

- putstring

  -
    ```
    @ Subroutine putstring: Provided a pointer to a null terminated string, putstring will
    @      display the string to the terminal
    @ R0: Must point to a null terminated string
    @ LR: Must contain the return address
    @ All registers are preserved.
    ```

- getstring

  -
    ```
    @ Subroutine getstring will read a string of characters terminated by a null
    @ R0: Points to first byte of buffer to receive the string
    @ R1: Maximum number of bytes in above buffer
    @ LR: Contains the return address

    @ Returned register contents:
    @ All registers are preserved.
    ```

You must link the library with your other object files after assembling your program.

i.e.

```
$ as -g -o rasm1.o rasm1.s
```

```
$ ld -o rasm1 rasm1.o barnett.a
```