Chapter
1

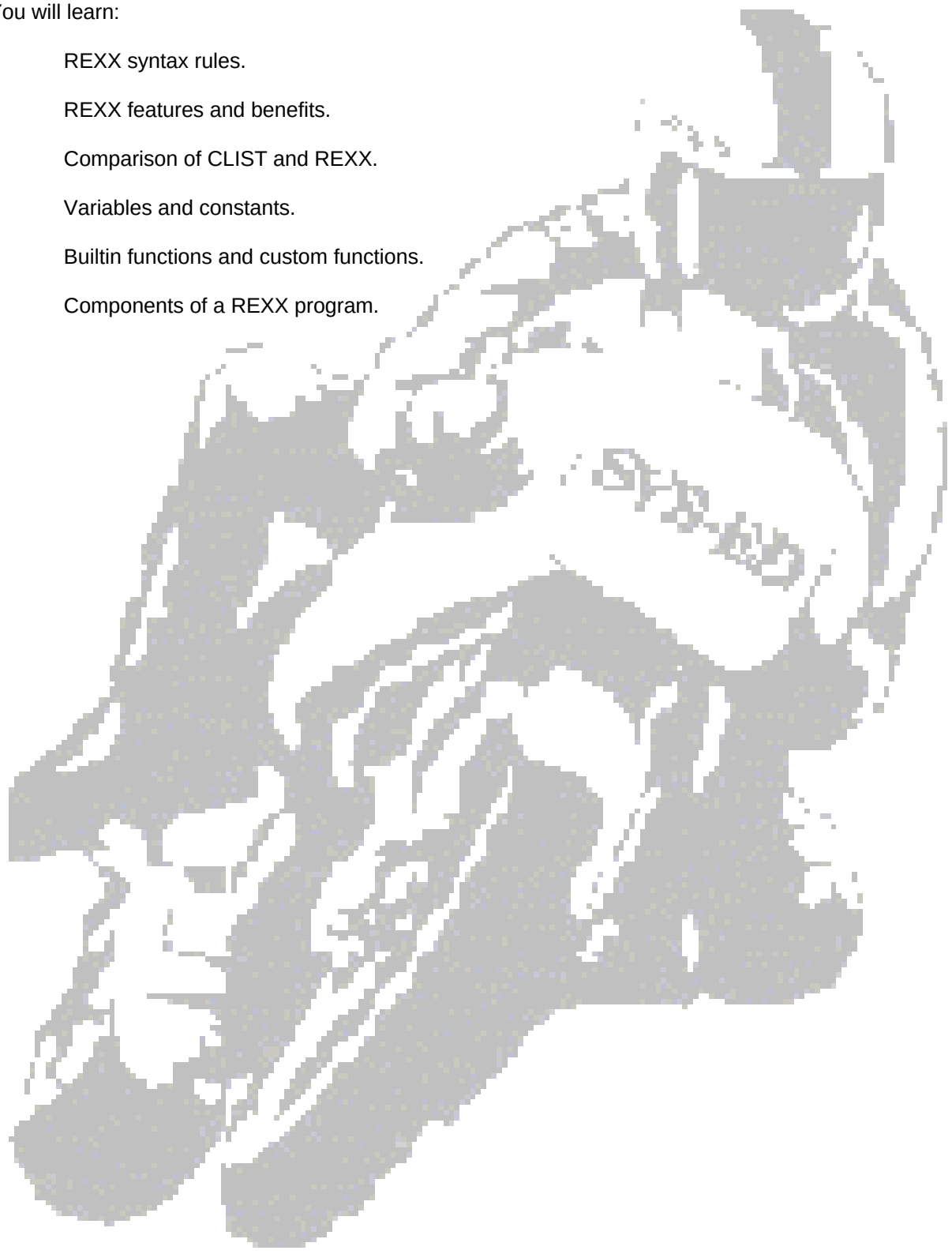
INTRODUCTION

**SYS-ED/
COMPUTER
EDUCATION
TECHNIQUES, INC.**

Objectives:

You will learn:

- REXX syntax rules.
- REXX features and benefits.
- Comparison of CLIST and REXX.
- Variables and constants.
- Builtin functions and custom functions.
- Components of a REXX program.



1 What is REXX?

REXX stands for Restructured eXtended eXecutor.

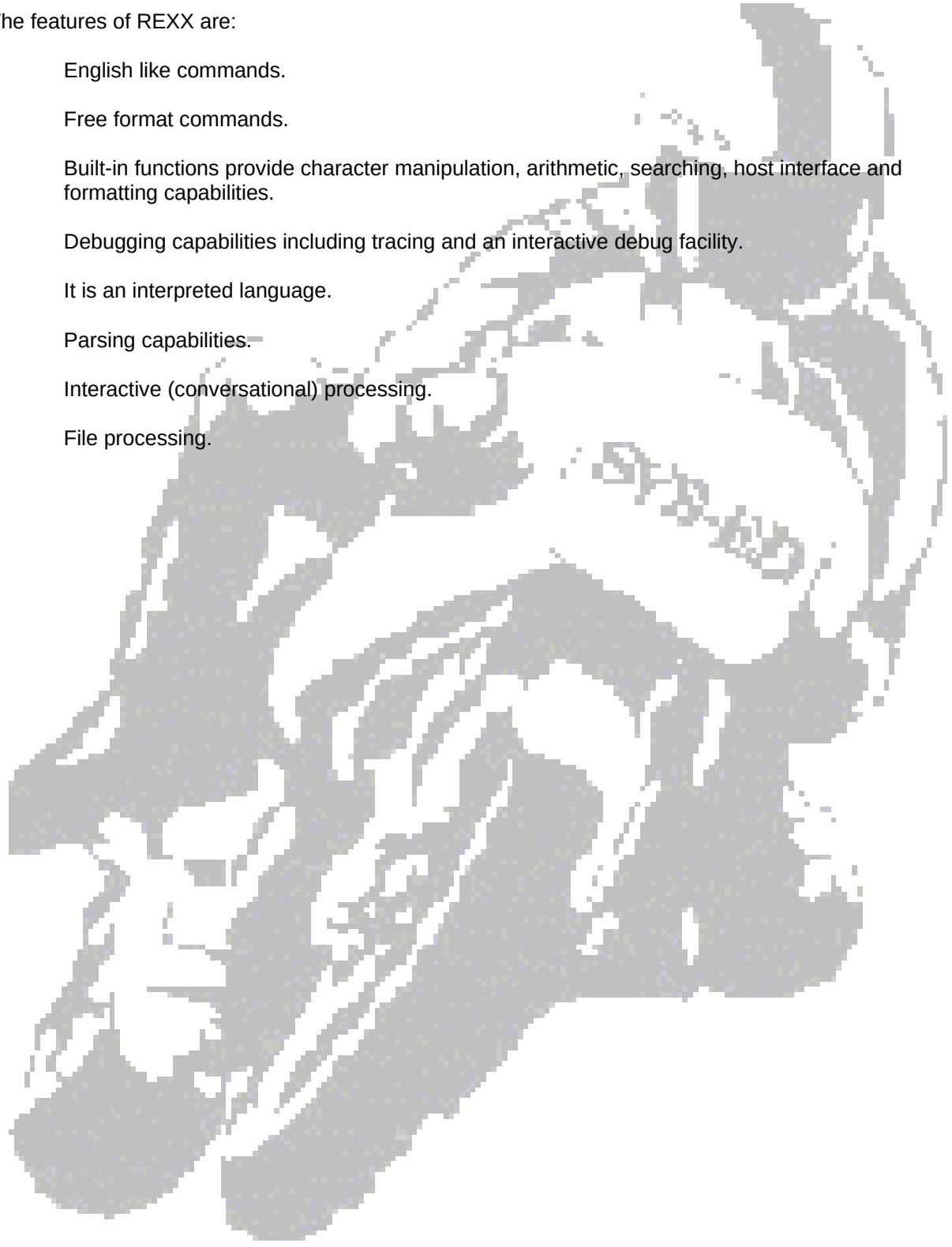
REXX:

- Creates command procedures.
- Is an implementation of the SAA procedure language.
- Provides a common programming structure.
- Includes character manipulation and arithmetic functions.
- Can be intermixed with host commands from TSO, CMS, CP, etc.
- Can run in a TSO/E or OS/390 and z/OS address space.

2 Features

The features of REXX are:

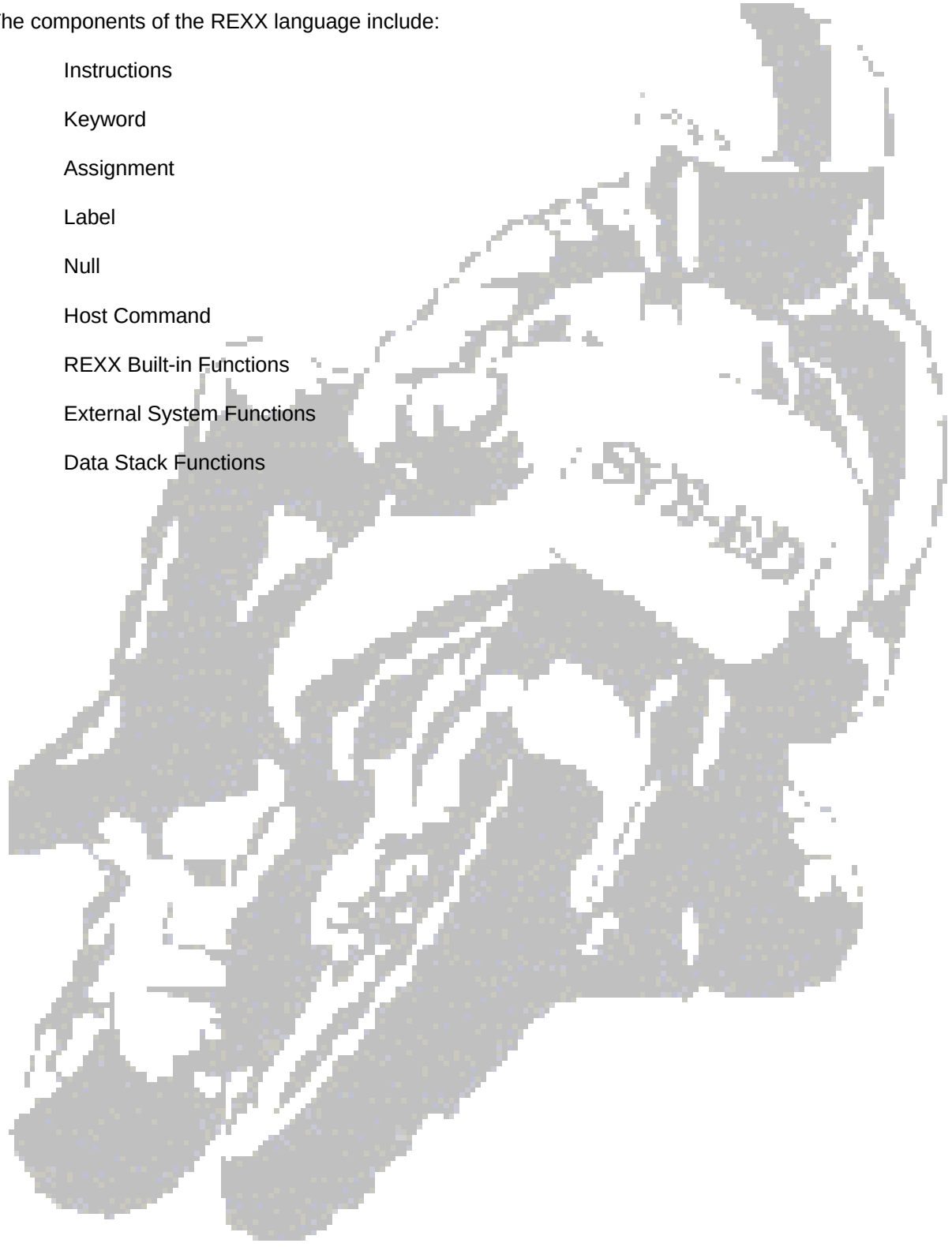
- English like commands.
- Free format commands.
- Built-in functions provide character manipulation, arithmetic, searching, host interface and formatting capabilities.
- Debugging capabilities including tracing and an interactive debug facility.
- It is an interpreted language.
- Parsing capabilities.
- Interactive (conversational) processing.
- File processing.



3 Components

The components of the REXX language include:

- Instructions
- Keyword
- Assignment
- Label
- Null
- Host Command
- REXX Built-in Functions
- External System Functions
- Data Stack Functions



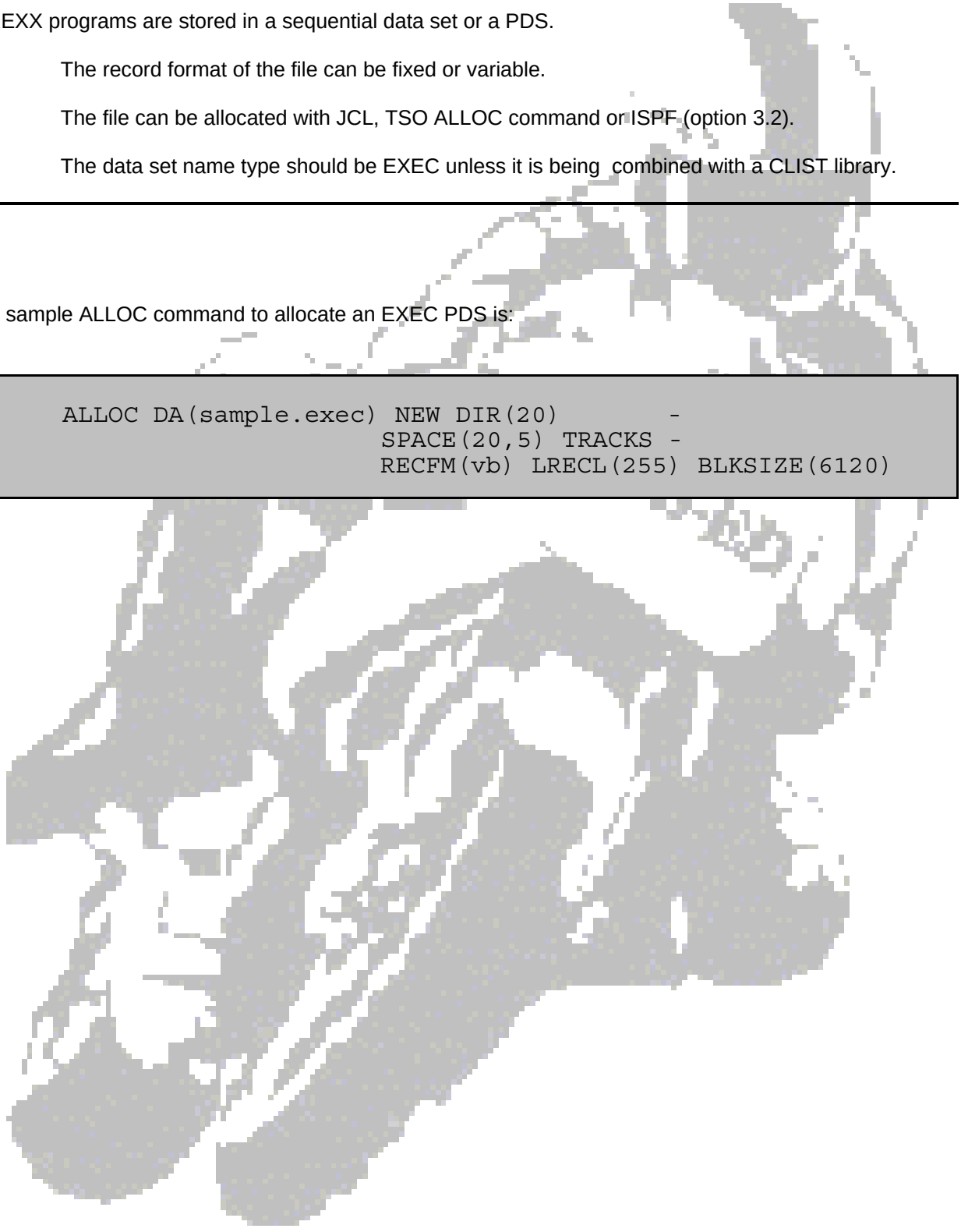
4 REXX Syntax

REXX programs are stored in a sequential data set or a PDS.

- The record format of the file can be fixed or variable.
- The file can be allocated with JCL, TSO ALLOC command or ISPF (option 3.2).
- The data set name type should be EXEC unless it is being combined with a CLIST library.

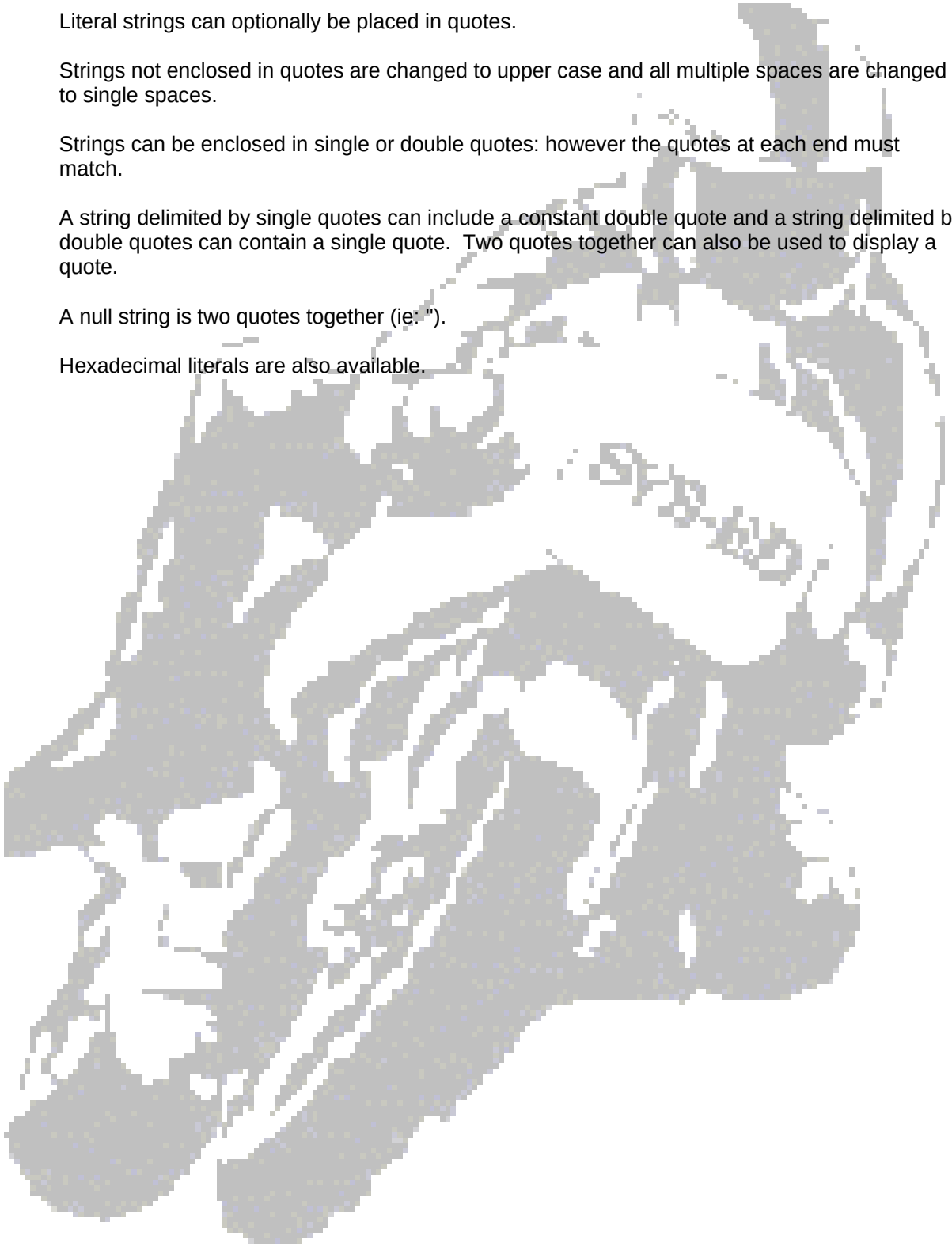
A sample ALLOC command to allocate an EXEC PDS is:

```
ALLOC DA(sample.exec) NEW DIR(20)       -
                      SPACE(20,5) TRACKS -
                      RECFM(vb) LRECL(255) BLKSIZE(6120)
```



5 Quotes and Strings

- Literal strings can optionally be placed in quotes.
- Strings not enclosed in quotes are changed to upper case and all multiple spaces are changed to single spaces.
- Strings can be enclosed in single or double quotes: however the quotes at each end must match.
- A string delimited by single quotes can include a constant double quote and a string delimited by double quotes can contain a single quote. Two quotes together can also be used to display a quote.
- A null string is two quotes together (ie: '').
- Hexadecimal literals are also available.



Examples:

SAY 'Dad, Can I Have The Car Tonight'

SAY "Dad, Can I Have The Car Tonight"

SAY Dad, Can I Have The Car Tonight

SAY 'Dad, Can I Have Mom"s Car Tonight'

SAY "Dad, Can I Have Mom's Car Tonight"

6 Continuation

- A REXX statement can begin in any column.
- It is preferable to have one statement per line.
- More than one statement can be put on a line, if a semicolon separates the commands.
- Statements that extend beyond one line can be continued by ending it with a comma and continuing on the next line.

Examples:

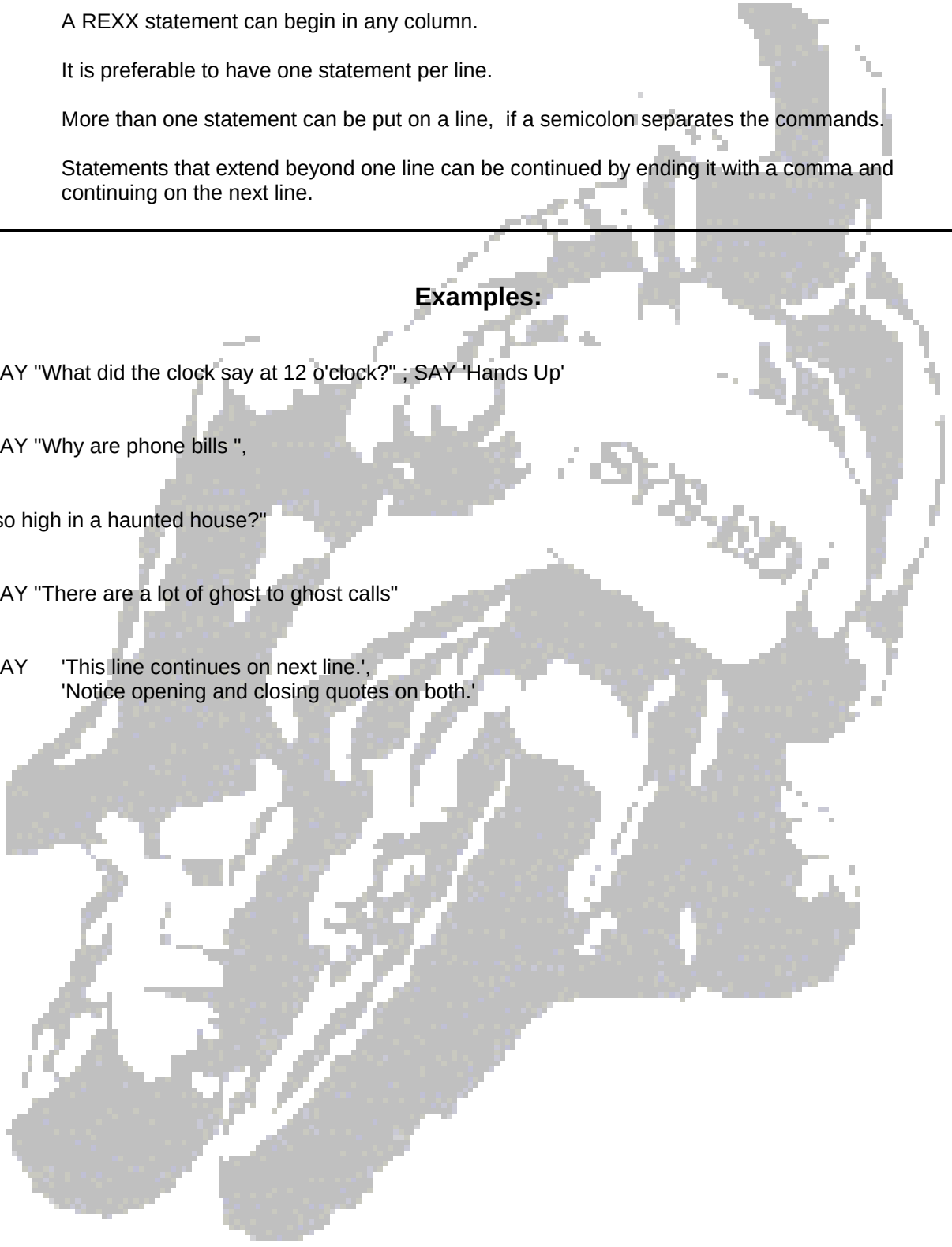
```
SAY "What did the clock say at 12 o'clock?" ; SAY 'Hands Up'

SAY "Why are phone bills ",

"so high in a haunted house?"

SAY "There are a lot of ghost to ghost calls"

SAY 'This line continues on next line.',
    'Notice opening and closing quotes on both.'
```



7 Types of Instructions

Keyword:

SAY, PULL, IF, DO etc.

Assignment:

total = total + amt

Label:

The label is used for subroutine processing and error checking. A form of the GOTO command is also available, but its use is discouraged.

A "read_loop:" is an example of a label.

Null:

A comment or a blank line used to improve readability.

Comments are string enclosed in a /* at the beginning and a */ at the end. A comment can span multiple lines.

All REXX procedures should have a comment on the first line that includes the word REXX. This is used by the system to distinguish between a CLIST and a REXX command procedure.

Command:

A command is a statement that is not a REXX keyword, assignment, null or a label.

Variables and expressions can be placed into a command.

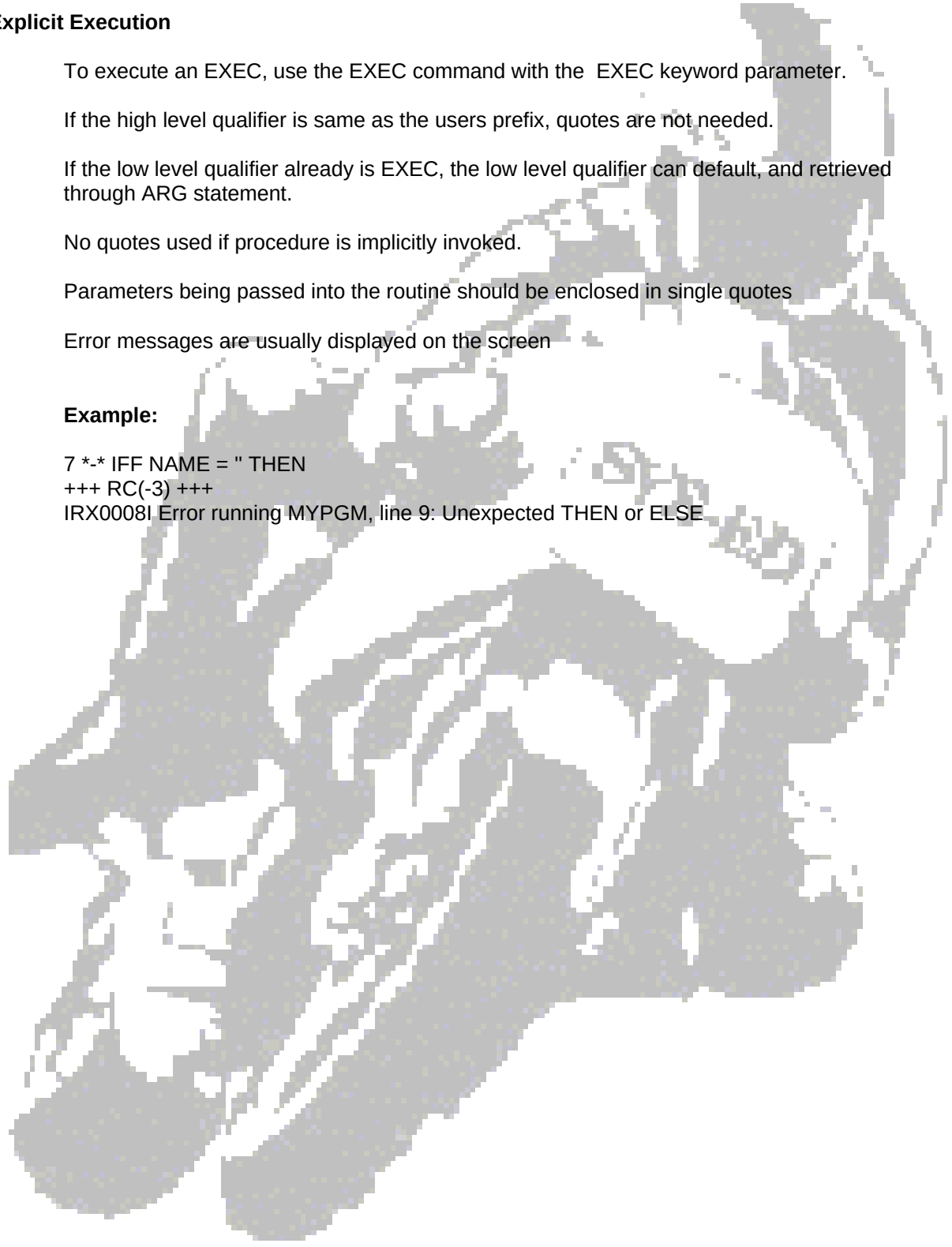
8 Executing a REXX Procedure

Explicit Execution

- To execute an EXEC, use the EXEC command with the EXEC keyword parameter.
- If the high level qualifier is same as the users prefix, quotes are not needed.
- If the low level qualifier already is EXEC, the low level qualifier can default, and retrieved through ARG statement.
- No quotes used if procedure is implicitly invoked.
- Parameters being passed into the routine should be enclosed in single quotes
- Error messages are usually displayed on the screen

Example:

```
7 *-* IFF NAME = '' THEN
+++ RC(-3) +++
IRX0008I Error running MYPGM, line 9: Unexpected THEN or ELSE
```



Examples:

EXEC 'ptc001.sample.exec(mypgm)' EXEC

EXEC sample.exec(mypgm) EXEC

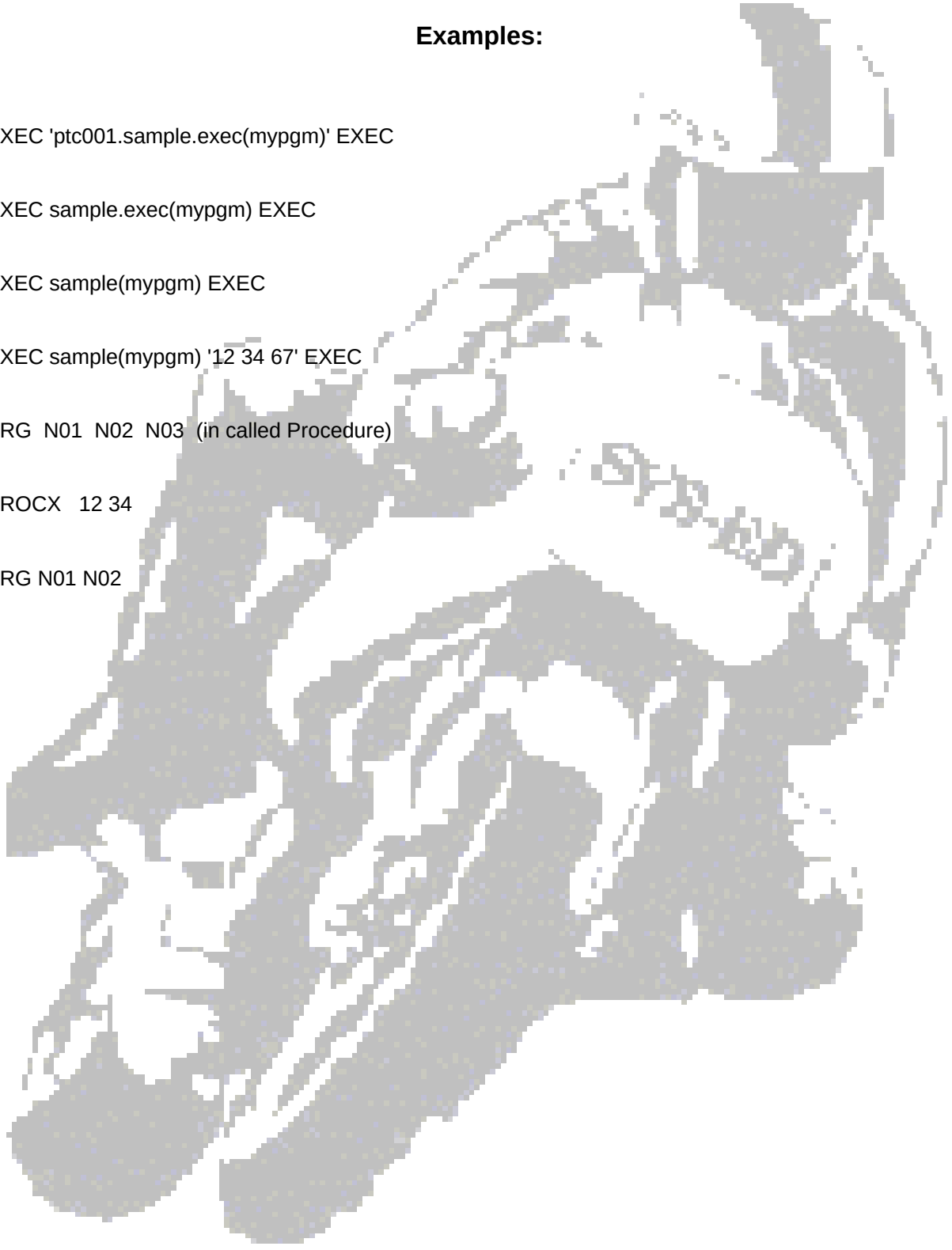
EXEC sample(mypgm) EXEC

EXEC sample(mypgm) '12 34 67' EXEC

ARG N01 N02 N03 (in called Procedure)

PROCX   12 34

ARG N01 N02



Implicit Execution

- Implicit execution allows you to run a REXX procedure by entering in the member name The format of standard TSO commands can be emulated..

- Before running an implicit procedure, a SYSPROC or SYSEXEC must be allocated with the EXEC libraries. Watch out for the order of concatenation.

- SYPROC can contain both CLISTs and REXX procedures.

  The REXX programs must have a comment in the first line with the word REXX on it.

- SYSEXEC can only contain REXX programs.

- SYSEXEC is searched before SYSPROC.

The following command will probably be need to be executed prior to the SYSEXEC being searched:

EXECUTIL SEARCHDD (YES)

Examples:

mypgm

mypgm 12 34 67

9 Allocation Procedures

```
/*  ALLOCATING SYSEXEC  */

"EXECUTIL SEARCHDD(YES)"

"ALLOC FI(SYSEXEC)  DATASET(MYSTUFF.EXEC) SHR REUSE"

IF  RC = 0        THEN SAY "SUCCESS ... WE GOT LIFTOFF"
                       ELSE SAY "ALLOCATION FAILED"

EXIT
```

```
/*        ALLOCATING SYSPROC        */

"ALLOC FI(SYSPROC) DATASET(MYSTUFF.exec",
                "         'SYS1.CLIST', ",
                "         'SYS1.CMDCLIST') SHR REUSE"

IF RC = 0                 THEN SAY "GOOD GOING"
                          ELSE SAY "GET GOING"

EXIT
```