```
; Name       : tree.asm
;
;build       : aclocal && autoconf && automake --add-missing --foreign
;              mkdir build
;              cd build
;              ../configure
;              make
;
;description : an example of glib 2.0 balanced binary trees
;
;source      : https://github.com/steshaw/gtk-examples

bits 64

[list -]
    extern    g_tree_destroy
    extern    g_tree_foreach    ;g_tree_traverse is deprecated since 2.2
    extern    g_tree_height
    extern    g_tree_insert
    extern    g_tree_lookup
    extern    g_tree_new
    extern    g_tree_nnodes
    extern    g_print
    extern    strcmp
    extern    exit
[list +]

%define NNODES    4             ;nodes to show in g_tree_foreach

section .rodata
    names:
    .fred:       db    "Fred",0
    .mary:       db    "Mary",0
    .sue:        db    "Sue",0
    .john:       db    "John",0
    .shelley:    db    "Shelley",0
    .mark:       db    "Mark",0
    .renato:     db    "Renato",0
    properties:
    .loud:       db    "Loud",0
    .obnoxious:  db    "Obnoxious",0
    .drunk:      db    "Drunk",0
    .quiet:      db    "Quiet",0
    .civil:      db    "Civil",0
    .strange:    db    "Strange",0
    .mighty:     db    "Mighty",0
    messages:
    .lookup:     db    "Looking up %s => value %s",10,0
    .height:     db    "Tree height: %d",10,0
    .nodes:      db    "Tree nodes: %d",10,0
    .tree:       db    "Tree:",10,0
    .node:       db    "key: %s %s value: %s",10,0
    userdata:    db    "=>",0

section .data

    tree:        dq    0        ;start of the tree
    flag:        db    0

section .text
global _start

_start:
    ;create tree with compare function
    mov     rdi,compare
    call    g_tree_new
    mov     [tree],rax
    ;insert the key/value pairs
    mov     rdi,[tree]
    mov     rsi,names.fred
    mov     rdx,properties.loud
    call    g_tree_insert
```

```
 73
 74      mov     rdi,[tree]
 75      mov     rsi,names.mary
 76      mov     rdx,properties.obnoxious
 77      call    g_tree_insert
 78
 79      mov     rdi,[tree]
 80      mov     rsi,names.sue
 81      mov     rdx,properties.drunk
 82      call    g_tree_insert
 83
 84      mov     rdi,[tree]
 85      mov     rsi,names.john
 86      mov     rdx,properties.quiet
 87      call    g_tree_insert
 88
 89      mov     rdi,[tree]
 90      mov     rsi,names.shelley
 91      mov     rdx,properties.civil
 92      call    g_tree_insert
 93
 94      mov     rdi,[tree]
 95      mov     rsi,names.mark
 96      mov     rdx,properties.strange
 97      call    g_tree_insert
 98
 99      mov     rdi,[tree]
100      mov     rsi,names.renato
101      mov     rdx,properties.mighty
102      call    g_tree_insert
103      ;search if Fred is in the list and print result
104      mov     rdi,[tree]
105      mov     rsi,names.fred
106      call    g_tree_lookup
107      mov     rdx,rax
108      mov     rdi,messages.lookup
109      xor     rax,rax
110      call    g_print
111      ;get tree height and print result
112      mov     rdi,[tree]
113      call    g_tree_height
114      mov     rsi,rax
115      mov     rdi,messages.height
116      xor     rax,rax
117      call    g_print
118      ;get tree nodes and print result
119      mov     rdi,[tree]
120      call    g_tree_nnodes
121      mov     rsi,rax
122      mov     rdi,messages.nodes
123      xor     rax,rax
124      call    g_print
125      ;print nodes 0 to NNODES
126      mov     rdi,messages.tree
127      xor     rax,rax
128      call    g_print
129      mov     rdi,[tree]
130      mov     rsi,display
131      mov     rdx,userdata
132      call    g_tree_foreach
133      ;destroy our tree and the pointer
134      mov     rdi,[tree]
135      call    g_tree_destroy
136      xor     rdi,rdi
137      mov     [tree],rdi                ;destroy pointer too
138      ;exit the program
139      xor     rdi,rdi
140      call    exit
141
142  display:
143      push    rbp
144      mov     rbp,rsp
```

```
145     mov     rcx,rsi
146     mov     rsi,rdi
147     mov     rdi,messages.node
148     xor     rax,rax
149     call    g_print
150     xor     rax,rax
151     inc     byte[flag]
152     cmp     byte[flag],NNODES          ;stop after n nodes
153     jl      .exit
154     inc     rax                        ;return TRUE
155 .exit:
156     mov     rsp,rbp
157     pop     rbp
158     ret
159
160 compare:
161     ;compare strings in rdi and rsi, returning
162     ;-1 when string rdi comes before string rsi
163     ; 0 when both strings are equal
164     ; 1 when string in rsi comes after string rdi
165     call    strcmp
166     ret
```