INSTALLING
- you'll need OpenGL and GLUT to compile and run it

COMPILING
for compiling the main heapdraw you'll need glut.

```
g++ main.cpp Heap.cpp draw.cpp -lglut
```

for compiling log-malloc.so:
gcc -nostartfiles -shared -fPIC -g -ldl -o log-malloc.so log-malloc.c

RUNNING

```
gera@poxiran$ ./heapdraw
Use: g [-a] [-q] [-v] [-p pid] [-r region] -R [lo-hi] -t type
        a       autoregions: automatically find regions
        q       quiet: don't print found errors
        v       verbose, more is more
        p       PID to follow, default is all, 0 is first, other is specific PID
        r       region to draw (0 is loose region, and it's the default)
        R       Manually define a region. lo and hi are its limits (can be hex)
        t       input file type, where type is one of:
                        ltrace
                                ltrace -o ls.ltrace -f -tt -i -
efork,vfork,clone,malloc,realloc,free,calloc,mmap,munmap,mremap,mprotect,mmap2 /bin/ls
                        truss
                                truss -o ls.truss -f -d -l -t !all -u
a.out,*::malloc,realloc,free,calloc,fork,vfork,mmap,munmap,mprotect,brk /bin/ls
                        ntsd
                                ntsd -cf hd.ini calc.exe
```

heapdraw will read input from stadard in, so you have to run it like:

```
$ heapdraw <vi.ltrace -t ltrace
```

When ran with -a, it will try to separate the heap in contiguos regions, this option is important mainly because if you draw a discontinuos heap where the holes are much bigger than the blocks, you won't see the blocks until you
zoom into the right spot. When using -a you'll also want to use -v to see how the regions where created and then -r to select what region to draw.

To the same end you can use -R to define a range by hand (format is -R 0x8048000-0x8090000). -R will create a region, but when drawing it will only use the ranges as far as there's any block, so defining a region like 0x8080000-0x9000000 is safe, and should not make everything look too small unless a lot of memory in this range is really used)

The option -p is usefull when you trace through several processes (fork()) and want to only see one process' heap. Have in mind that when selecting a child process, heapdraw will not follow the heap in the parent until it forks, so you will not see the initial state of the heap for the child process.

For graphical hotkeys usage see main README

On linux we usually use ltrace or a preloaded malloc/free hook library on some weird cases. On Solaris we use truss and on windows we used to use an ntsd script that we lost :-(

log-malloc.so must be used as:

```
$ LD_PRELOAD=../log-malloc.so ls -lR .. 2> ls-lr.ltrace
```

or, if you don't want to use stderr, you can rather do:

```
$ LOG_MALLOC_OUTPUT=ls-lr.ltrace LD_PRELOAD=./log-malloc.so ls -lR ..
```

if you prefer to use ltrace (for example to attach to a running process
after some initialization has already happened):

```
$ ltrace -o ls.ltrace -f -tt -i -
efork,vfork,clone,malloc,realloc,free,calloc,mmap,munmap,mremap,mprotect,mmap2 /bin/ls
```

or, of course:

```
$ ltrace -o ls.ltrace -f -tt -i -
efork,vfork,clone,malloc,realloc,free,calloc,mmap,munmap,mremap,mprotect,mmap2 -p `pidof
-s bash`
```

For Solaris or Windows instructions, follow the usage help of heapdraw.