

```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
```

```
#define MAXPAROLA 30
#define MAXRIGA 80
```

```
int main(int argc, char *argv[])
```

```
{
    int freq[MAXPAROLA]; /* vettore di contatori
delle frequenze delle lunghezze delle parole */
    char riga[MAXRIGA];
    int i, inizio, lunghezza;
    FILE * f;
```

```
    for(i=0; i<MAXPAROLA; i++)
        freq[i]=0;
```

```
    if(argc != 2)
```

```
    {
        fprintf(stderr, "ERRORE: serve un parametro con il nome del file\n");
        exit(1);
    }
```

```
    f = fopen(argv[1], "r");
    if(f==NULL)
```

```
    {
        fprintf(stderr, "ERRORE: impossibile aprire il file %s\n", argv[1]);
        exit(1);
    }
```

```
    while( fgets( riga, MAXRIGA, f ) != NULL )
```

Operating Systems gdb tutorial

Compiling

`gdb` is a line interface debugger for C (and C++).

- ❖ To prepare a program for debugging with **`gdb`**, you must compile it with the **`-g`** flag. Example:

```
gcc -g -o myprog myprog.c
```

Running and quitting gdb

To debug your program, run

```
> gdb myprog  
(gdb)
```

To quit debugging your program, give command
quit (or just q)

Command help

help displays a list of topics (classes of commands).

.....

breakpoints -- Making program stop at
 certain points

data -- Examining data

files -- Specifying and examining files

.....

Command help

help topic displays information about that topic
(gdb) help breakpoints

help command displays information about a
specific command
(gdb) help print

Command run

`run (r)` run the executable given as argument to **`gdb`**.

`run args` run the executable given as argument to **`gdb`**, with the arguments that you would pass in the command line

`r arg1 arg2 ... argn`

Input/output redirection is possible

`r > outfile.txt`

Command break

break **linenumber** **or**

break **filename:linenumber**

sets the breakpoint to the given line number in the source file.

Execution will stop before that line has been executed.

break **(b)** **function** **or**

break **function:linenumber**

sets the breakpoint at the **linenumber** of function

Command delete and info

delete deletes all breakpoints.

delete number deletes breakpoint number
number

info breakpoints

shows all current breakpoints, including their
number

Commands continue, next, step

continue (c)

continues the program execution, after the breakpoint

next (n)

executes the next instruction (function)

step (s)

steps into the first instruction of a function

Command list

list(l)linenumber

displays 10 lines from the source code around
linenumber.

list(l) function

displays 10 lines from the beginning of
function

list(l)

displays the next 10 lines

Commands print

print (p) expression

displays the value of **expression**.

print v[0]@5

displays the first 5 values in array **v**