This system call is called ‘getdents’ (see System calls link).
- its number is 141 (put it in eax).
- ebx should contain file descriptor:
  each directory has a file with its files names and some other information; file name in this case is the (full) path to the directory from the root; we should open this file with O_RDONLY (with pmode = 0)
- ecx contains a pointer to the input buffer (it should be pointer of type ent of the following structure)
- edx contains a number of bytes to read (it’s enough to reed 100 bytes since you have discover only two first file names)

```
typedef struct ent {
    int inode;
    int offset;
    short len;
    char buf[1];
    } ent;
```

getdents			141
	get directory entries		
arg	eax	141	
	ebx	uint fd	
	ecx	struct dirent *dirp	
	edx	uint count	
return	eax	no. of bytes read	
errors		badf, fault, inval, noent, notdir	
ref	unistd.h, linux/dirent.h, linux/unistd.h		

NOTE: False description of structure <dirent> in man 2 de getdents:
2nd item, "d_off", is **offset from beginning of directory file** to concerning entry.

Here is the code that:

- get the name of the current directory
- open the current directory file
- read it into an array of ent structures

```
char buf[2000];
int fd;
ent *entp = buf;
int count;

// get (full) path to the current directory
getwd(wd);

printf("Getcwd returns %s\n", wd);

// open current directory file
fd = system_call(5,0,0,0)
if(fd < 0)
        exit(1);

printf("Got fd %d\n", fd);
count = system_call(141,fd,buf,100)
```

The returned value of the second 'system_call' is a number of bytes that were read.

Let's examine the structure more closely:

```
typedef struct ent {
    int inode;
    int offset;
    short len;      ← length of the current record
    char buf[1];    ← pointer to the file name
    } ent;
```

Assume that your directory contains only one file called *assign3.html*.

After you execute 'getdents' system call (that reads the content of the current directory), entp (and buf) points to the following array of records:

```
Inode is 5279235, offset is 1, size 12, name .
Inode is 5279236, offset is 2, size 16, name ..
Inode is 5279237, offset is 3, size 24, name assign3.html
```

First two records describe current directory and the parent directory.
All other records describe files in the current directory.

Notes:
1. Each file name terminates with Null, that is added to a size of a file name.
2. Each record length should be 0 mod 4, so it is completed with empty bytes if needs.

After reading the current directory, we would like to get name of the first two files in it. How would we do this?

In the following way: since the first two records are always the same, we can just skip it (add 28 bytes to entp pointer).
We will read the third record (the record of the first file in the current directory), save the name this file and move the length of this record in order to read the second record:

```
entp = buf+28;

printf("Inode is %d, offset is %d, size %d, name %s\n",
        entp->inode,
        entp->offset,
        entp->len,
        entp->buf);
```

```
Inode is 5279237, offset is 3, size 24, name assign3.html
```

Now we know that the size of this record is 24, so we can jump to the next record by adding 24 to entp. Then we can read the next file record at the same way.

Note: don’t forget to close the directory file!