获取环境变量

相关函数：putenv, setenv, unsetenv

头文件：#include <stdlib.h>

定义函数：char * getenv(const char *name);

函数说明：getenv()用来取得参数name 环境变量的内容. 参数name 为环境变量的名称, 如果该变量存在则会返回指向该内容的指针. 环境变量的格式为name＝value.

返回值：执行成功则返回指向该内容的指针, 找不到符合的环境变量名称则返回NULL.

范例

```
#include <stdlib.h>
main()
{
    char *p;
    if((p = getenv("USER")))
    printf("USER = %s\n", p);
}
```

执行：

```
USER = root
```

设置环境变量

头文件：#include<stdlib.h>

定义函数：int putenv(const char * string);

函数说明：putenv()用来改变或增加环境变量的内容. 参数string 的格式为name＝value, 如果该环境变量原先存在, 则变量内容会依参数string 改变, 否则此参数内容会成为新的环境变量.

返回值：执行成功则返回0，有错误发生则返回-1.

错误代码：ENOMEM 内存不足，无法配置新的环境变量空间.

范例

```
#include <stdlib.h>
main()
{
    char *p;
    if((p = getenv("USER")))
    printf("USER =%s\n", p);
    putenv("USER=test");
    printf("USER+5s\n", getenv("USER"));
}
```

执行：

```
USER=root
USER=root
```

清除环境变量

unsetenv()清除某个特定的环境变量的函数。

另外，还有一个指针变量environ，它指向的是包含所有的环境变量的一个列表。下面的程序可以打印出当前运行环境里面的所有环境变量：

实践例子：（env.c）

```
1. #include <stdio.h>
2. extern char**environ;
3. int main ()
4. {
5.     char**var;
6.     char *str;
```

```
7.     for (var =environ;*var !=NULL;++var)
8.     printf ("%s/n",*var); //output all env
9.     str = (char *)getenv("CROSS_COMPILE"); //get CROSS_COMPILE env
10.     printf ("/n/n#####################################/n/n");
11.     printf ("CROSS_COMPILE = %s/n",str);
12.     return 0;
13. }
```