Environment variables ....

int main (int argc, char **argv, char **envp)

- Array of strings
- Each string of form "name=value"
- Last entry is NULL
- Examples:
  - HOME=/home/phil
  - CVSROOT=/home/phil/CVSrep

- int execve(const char *path, char *const argv[], char *const envp[])  
- int execvp(const char *path, char *const argv[])  
  - Access via "extern char **environ;"
  - (*BSD, man environ(7))
Program access to environment variables

- Access via "extern char **environ;"
- Library calls access ... easier
  - char *getenv(const char *name)
    - standard on all UNIX systems
    - returns a pointer to value in array
    - Do not free() returned pointer.
    - Depending on use, use strdup(3).
- int setenv(const char *name, const char *value, int overwrite);
- int putenv(const char *string);
- void unsetenv(const char *name);
- *BSD standard and linux
Changing Directories

- **chdir(2)**
- **int chdir(const char *path);**
  - sets the working directory

- **Return Values**
  - 0 if successful
  - -1 if failed, errno has error
Expand

function to start more processing on your input line in your ush

- int expand (char *line, char *new, int newsize)
  - line: the original line
  - new: a pointer to a new character array for "output"
  - newsize: the physical size of the "new" array
    - line has the End-of-string marker, new doesn’t

- expansions: copy characters from line to new except the following
  - ${NAME} -- all characters replaced by value of environment variable NAME
  - $$ -- replace the $$ with the ASCII representation of the shell’s PID
  - snprintf(3) is a great tool to use for this
Code outline

- Code should do ONE pass of the original line.
- Outline: (for clarity, not necessarily for your code)

  current character is first character of line
  while not at end of string and space is available in new
  if current character $
    if ${ or $$
      process expansion
    else
      copy $ to new and move to next character
  else
    copy current character to new and move to next character

Add END-OF-STRING to end of the "new" string and return success
or print overflow message if no room left in new and return failure