UCSC Extension – Linux System Programming
Homework assignment #3
Jae Yang Park (jaeyangp@gmail.com)

1.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
	printf("PATH = %s\n", getenv("PATH"));

	return 0;
}
```

```
output:
PATH =
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/sn
ap/bin
```

2.
2.1

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>

int main(int argc, char *argv[])
{
	pid_t my_pid;

	my_pid = getpid();

	printf("Program name = %s, pid = %d\n", argv[0], (int)my_pid);

	execl("/bin/ls", "/bin/ls", NULL, NULL);

	return 0;
}
```

2.2

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>

int main(int argc, char *argv[])
{
	pid_t my_pid;

	my_pid = getpid();

	printf("Program name = %s, pid = %d\n", argv[0], (int)my_pid);

	char *temp[] = { "/bin/ls", "-l", ".", NULL };
	execv("/bin/ls", temp);

	return 0;
}
```

3.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>

int main(int argc, char *argv[])
{
        pid_t my_pid;

        my_pid = getpid();

        printf("Program name = %s, pid = %d\n", argv[0], (int)my_pid);

        char *temp[] = { "/bin/ps", "-f", NULL };
        execv("/bin/ps", temp);

        return 0;
}
```

4.

```
include <stdio.h>
#include <unistd.h>

int main()
{
        printf("Maximum number of processed forked = %ld\n", sysconf(_SC_CHILD_MAX));
}
```

output:

```
Maximum number of processed forked = 14636
```

5.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main()
{
        pid_t pid;
        int i;

        pid = fork();

        if (pid < 0) {
                perror("fork failed");
                return -1;
        }

        if (pid == 0) {
                for (i = 0; i < 5; i++) {
                        sleep(10);
                        printf("(pid: %d) Hello World!\n\n", (int)getpid());
                }
        } else {
                exit (1);
        }
```

```
      return 0;
}
```

6.

6.1

```
// #1. Global un-initialized array
#include <stdio.h>
int myarray[50000];

int main(void)
{
      myarray[0] = 1;
      return 0;
}
```

```
   text          data          bss          dec          hex     filename
   1099           544       200032       201675        313cb     hw03_6a
```

```
// #2. Global initialized array
#include <stdio.h>
int myarray[50000] = {1};

int main(void)
{
      myarray[0] = 1;
      return 0;
}
```

```
   text          data          bss          dec          hex     filename
   1099        200560            8       201667        313c3     hw03_6b
```

```
 // #3. Stack un-initialized array
#include <stdio.h>

int main(void)
{
      int myarray[50000];
      myarray[0] = 1;
      return 0;
}
```

```
   text          data          bss          dec          hex     filename
   1256           552            8         1816          718     hw03_6c
```

```
// #4. Stack initialized array
#include <stdio.h>

int main(void)
{
      int myarray[50000] = {1};
      myarray[0] = 1;
      return 0;
}
```

```
   text          data          bss          dec          hex     filename
   1361           560            8         1929          789     hw03_6d
```