Fork process graphs

You are asked to answer questions about 10 sample programs which make use of fork(). Assume the programs run normally. For each program, first construct a process graph by hand.

This assignment is an extension of the previous assignment providing more hands-on experience with fork() calls and process graphs. You could even try building a tool to construct a process graph or identify a tool that automatically constructs a process graph given a C program.

program: p0.c

How many processes are created upon the execution of this program in the shell (including the first process created by the shell)?

```
#include <sys/types.h>
#include <unistd.h>
int main() {
   pid_t pid;

   pid = fork();
   if (pid != 0) {
      fork();
   }
   fork();
   return 0;
}
```

program: p1.c

How many processes are created upon the execution of this program in the shell (including the first process created by the shell)?

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

int main() {
   pid_t pid;

   fork();
   pid = fork();
   if (pid == 0) {
      fork();
   }
   return 0;
}
```

program: p2.c

What is the total number of processes that will be created by running this program? How many of each character ‘A’ to ‘C’ will be printed out?

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
int main() {
   pid_t pid1, pid2, pid3;

   pid1 = fork();
   pid2 = fork();

   if (pid1 == 0) {
      printf("A");
   } else {
      pid3 = fork();
      printf("B");
   }

   if (pid2 != 0 && pid3 != 0) {
      printf("C");
   }
}
```

program: p3.c

What is the total number of processes that will be created by running this program?

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
int main() {
   pid_t pid1, pid2;

   pid1 = fork();
   pid2 = fork();

   if (pid1 != 0 && pid2 != 0) {
      printf("A");
   }

   if (pid1 != 0 || pid2 != 0) {
      printf("B");
   } else {
      printf("C");
   }
}
```

program: p4.c

How many times will the message “Hello!” be displayed?

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

int main() {
   fork();
   fork();
   if (fork() == 0) {
      printf("Hello!");
   } else {
      printf("Goodbye!");
   }
}
```

program: p5.c

List all of the possible outputs that can occur when this program is run.

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

int main() {
   puts("0\n");
   if (fork() != 0) {
      puts("1");
      fork();
      puts("2");
   } else {
      puts("3");
   }
}
```

program: p6.c

List all the possible outputs that can occur when this program is run.

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

int main() {
   fork();
   if (fork())
      puts("Hello");
   if (fork())
      puts("Hello");
}
```

program: p8.c

Suppose I want to print exactly 37 exclamation marks (!). How many successive calls would I need to write in place of –X—to produce this output?

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

int main() {
   puts("!");
   if (fork()) {
     puts("!");
   }
   else {
     // -- X --
     puts("!");
   }
   puts("!");
   puts("!!");
}
```