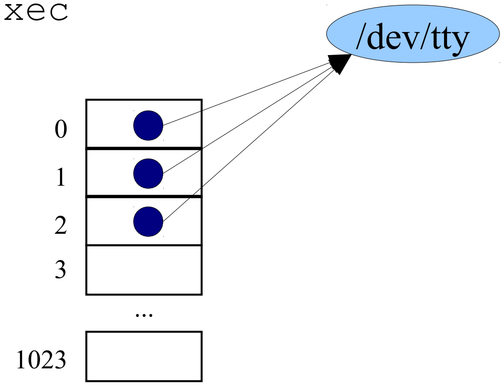
Princeton University / COS 217 / Trace of testdupforkexec

```
% gcc217 testdupforkexec.c -o testdupforkexec
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```

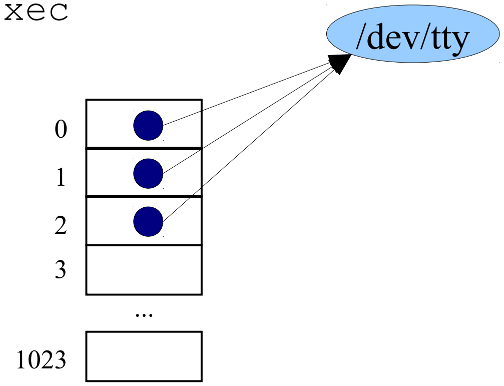


```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

...

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```

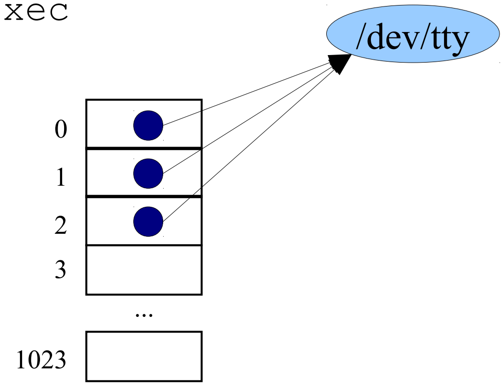


```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

...

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

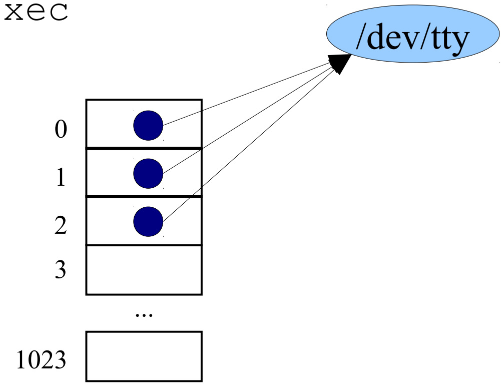
Writes to stdout (alias /dev/tty):

```
1140 parent
```

...

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```

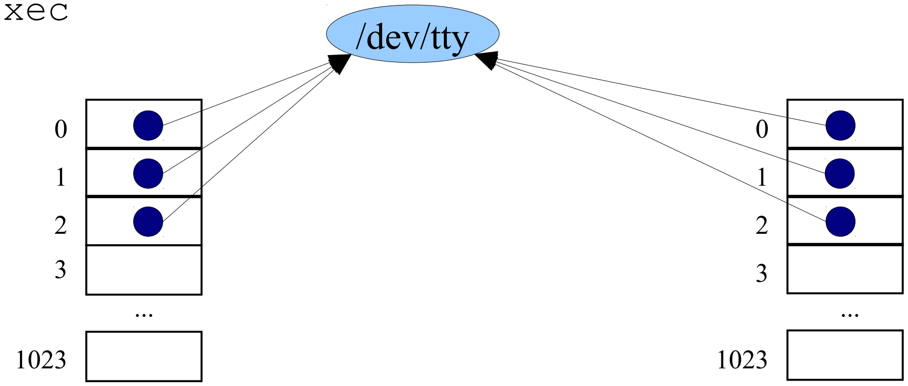


```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

...

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```

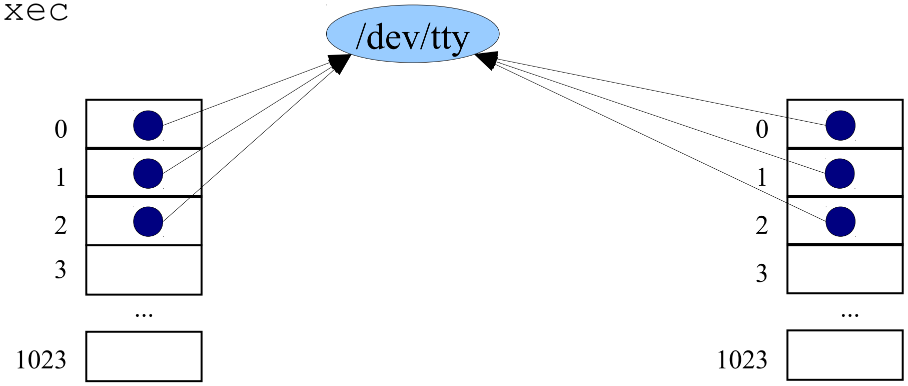


```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



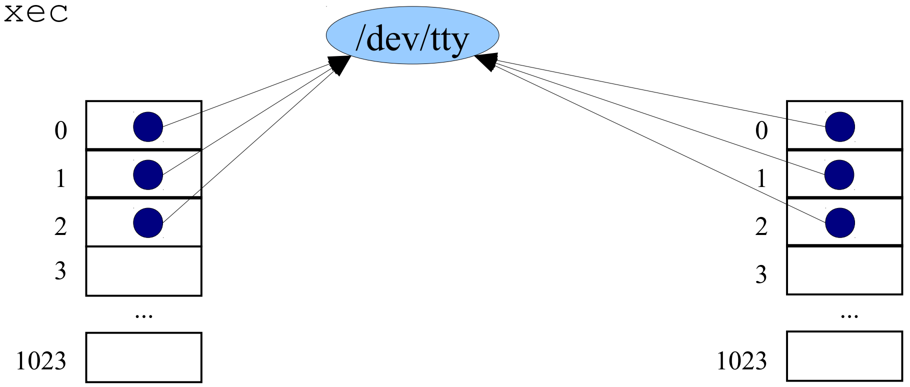
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



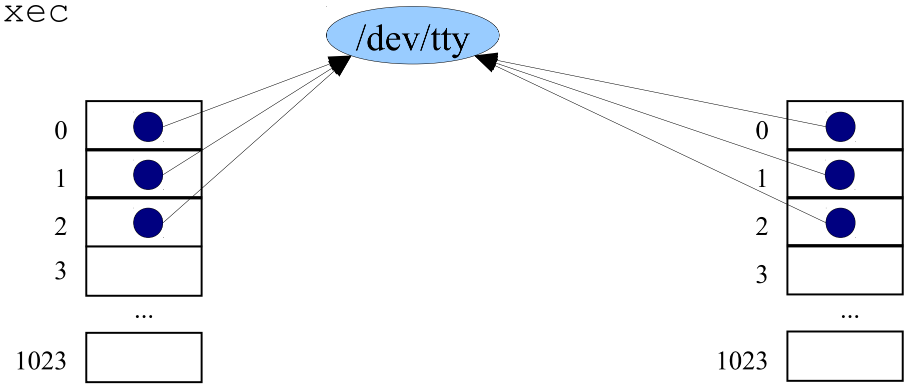
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



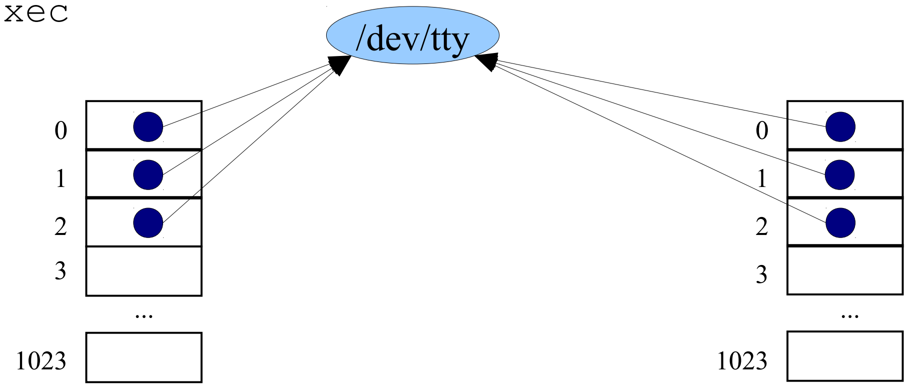
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

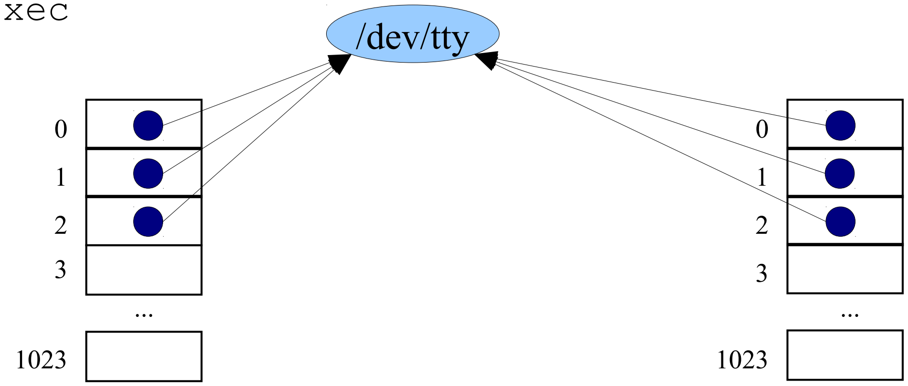
non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

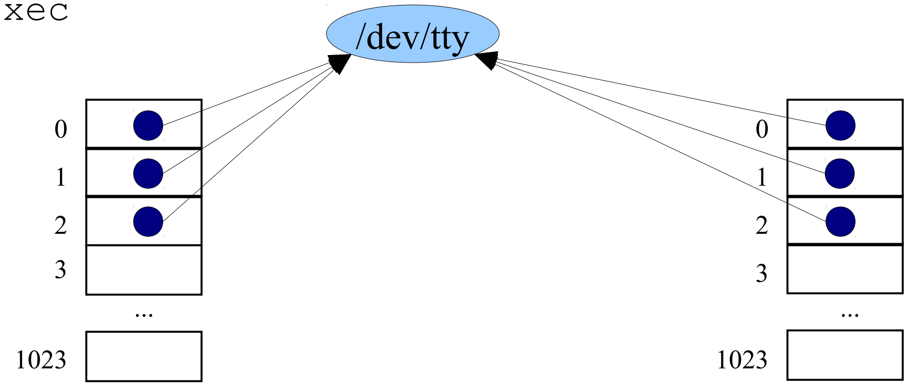
non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

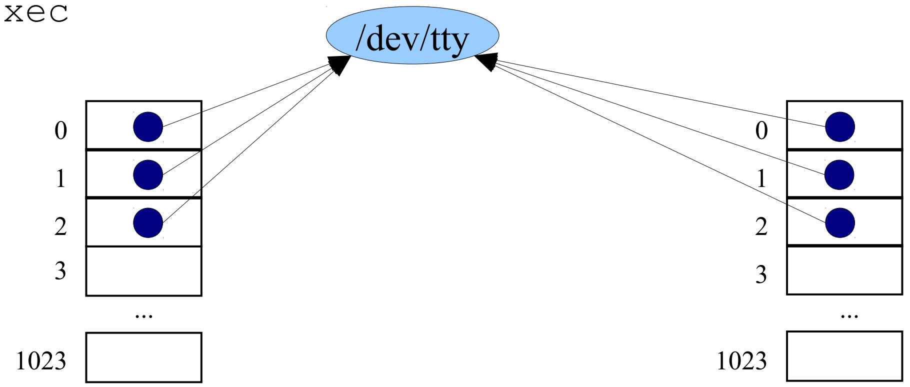
non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

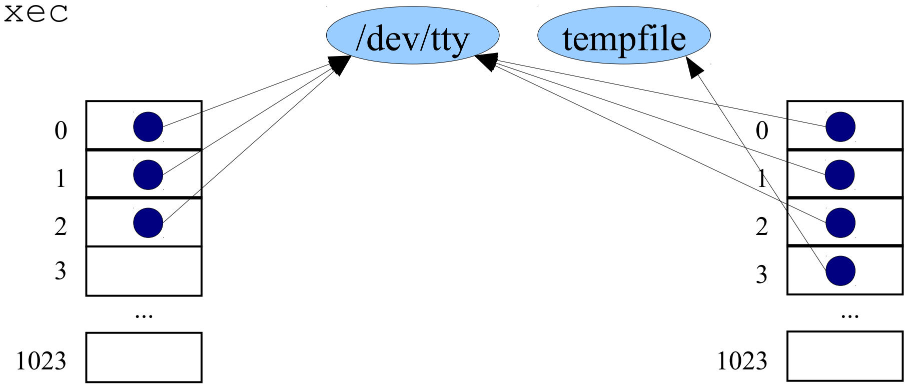
non-zero

```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

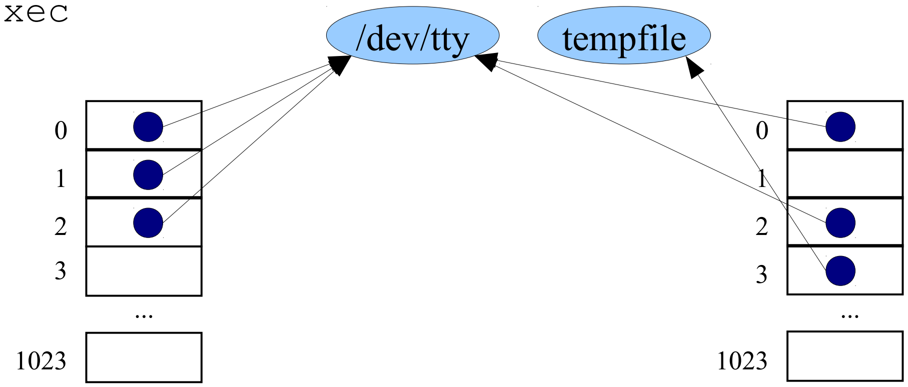
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

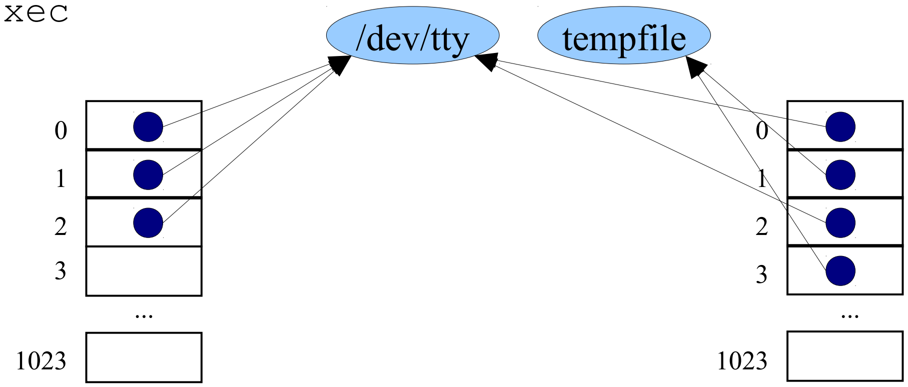
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

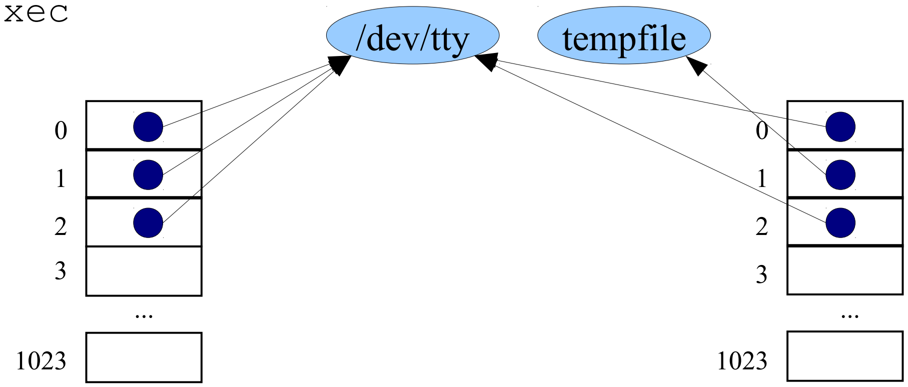
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

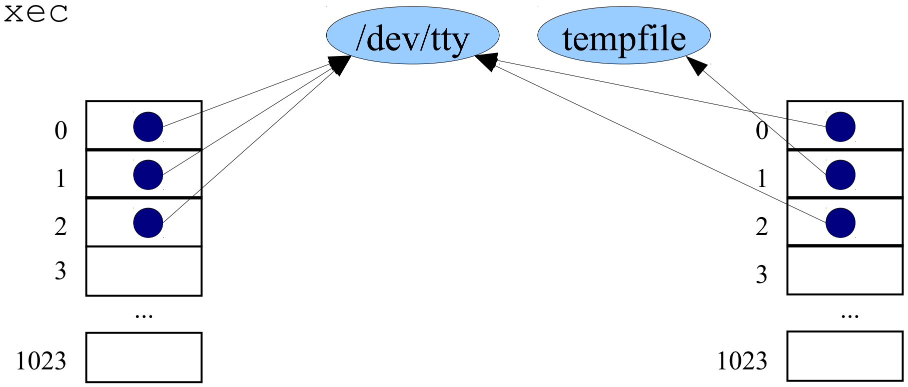
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

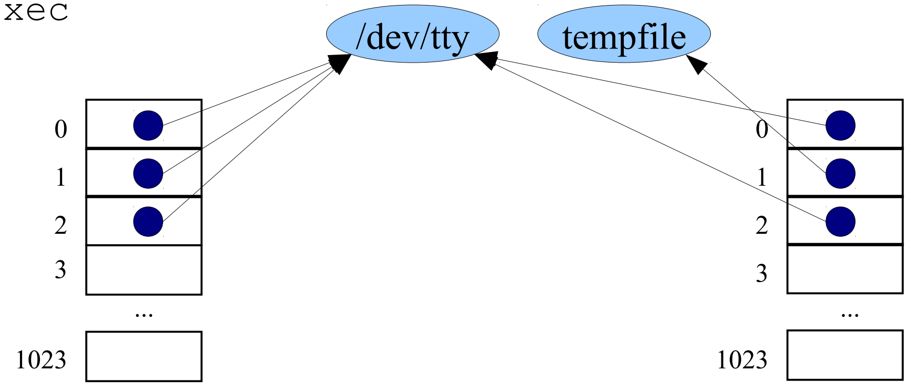
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

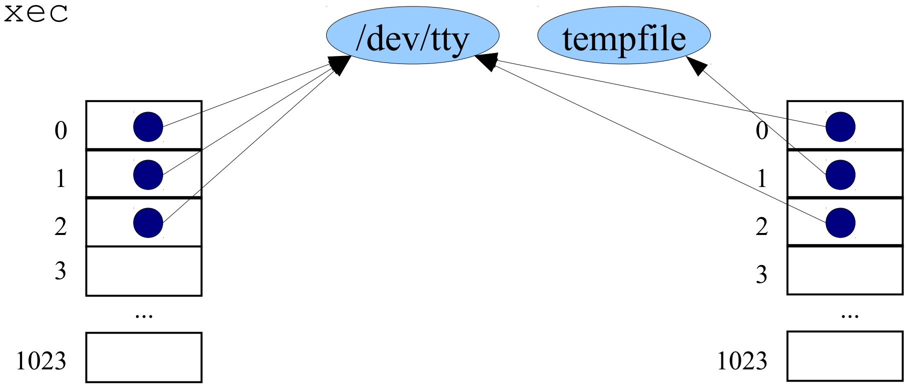
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

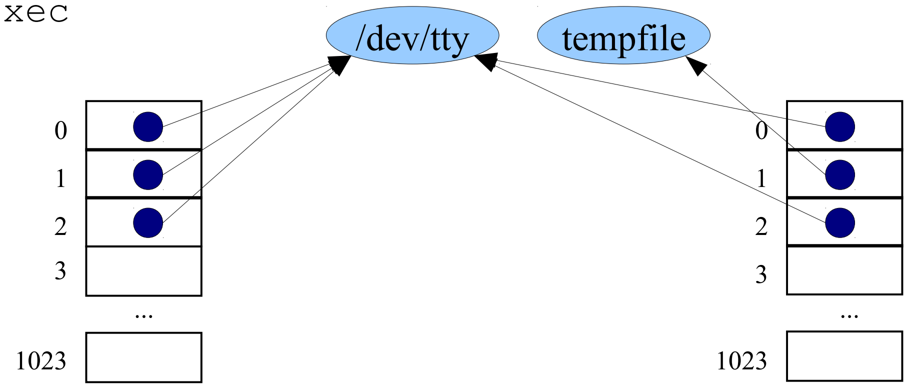
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

0

3

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



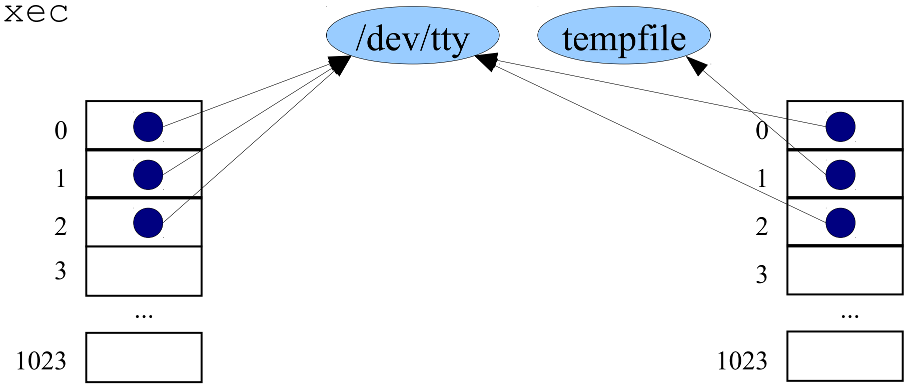
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

```
int main(int argc, char *argv[])
{

               Date
              program



   return 0;
}
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

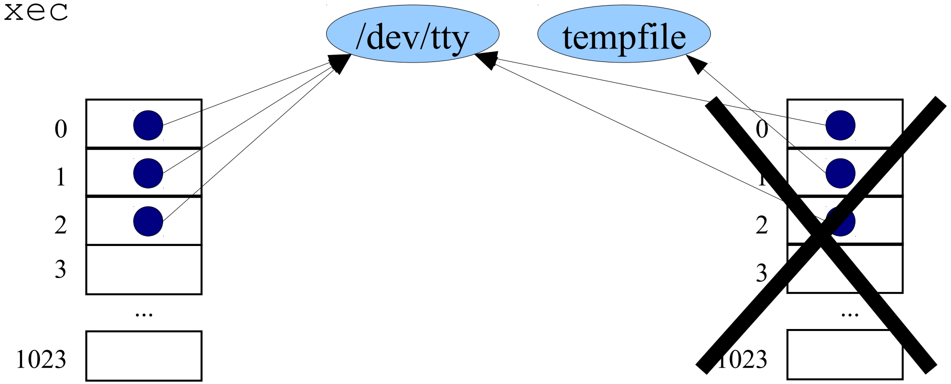
non-zero

```
int main(int argc, char *argv[])
{

            Date
           program


   return 0;
}
```

Writes the current date/time to stdout (alias tempfile)

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



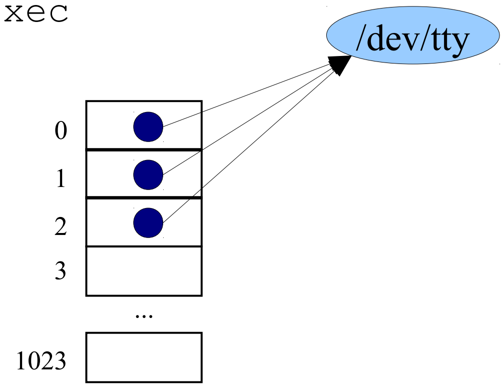
```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

```
int main(int argc, char *argv[])
{

               Date
              program



   return 0;
}
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

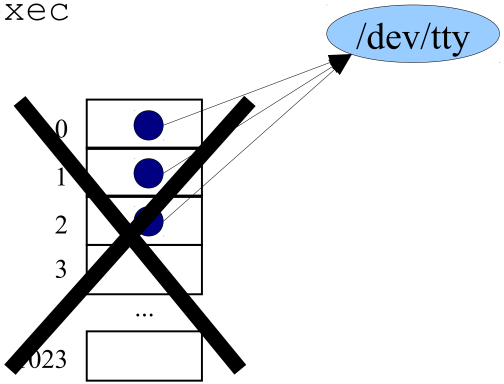
non-zero

Writes to stdout (alias /dev/tty):

```
1140 parent
```

Princeton University / COS 217 / Trace of testdupforkexec

```
% ./testdupforkexec
```



```
int main(int argc, char *argv[])
{  pid_t iPid;
   printf("%d parent\n",
      (int)getpid());
   fflush(NULL);
   iPid = fork();
   if (iPid == 0)
   {  char *apcArgv[2];
      int iFd;
      iFd = creat("tempfile",0600);
      close(1);
      dup(iFd);
      close(iFd);
      apcArgv[0] = "date";
      apcArgv[1] = NULL;
      execvp("date", apcArgv);
      perror(argv[0]);
      exit(EXIT_FAILURE);
   }
   wait(NULL);
   printf("%d parent\n",
      (int)getPid());
   return 0;
}
```

non-zero

```
%
```