1. Se realiza el código adjunto. Realiza un diagrama con los procesos creados. Explica a través del diagrama cómo finalizan los procesos, si quedan procesos zombies o huérfanos y cuáles son si los hubiese.

```
main(){
    fork();
    fork();
    fork();
    wait();
}
```

2. Realiza un diagrama con los procesos creados a partir de la ejecución del siguiente código. Identifica cada proceso con una letra e indica lo que imprimirán por la terminal cada uno de los procesos. Indica también si quedan procesos zombies o huérfanos y si se puede determinar cuáles.

```
main(){
    static int a[2]={0,0};
    static int c = 0;

    if(c == 3) exit(0);
    ++c;

    if (fork()) ++a[0];
    else ++a[1];
    main();

    printf ("(%d,%d)",a[0],a[1]);

    if(c != 3) wait();
}
```

3. Dibuja el árbol de procesos y determina qué procesos quedan zombies o huérfanos para el siguiente código:

```
main(){
    static int i=0;

    if(i != 3){
        ++i;
        fork();
        main();
    } else wait();
    exit(0);
}
```

4. Dibuja un esquema en forma de árbol de los procesos que se crean según el código que se indica más abajo. Explica también si los procesos terminan correctamente o no lo hacen y por qué motivos.

```
main(){
    for (i=0; i< 6; ++i)
        if(fork()) break;
    if(i<5) wait();
    exit();
}
```

5. Se realiza el código adjunto. Realiza un diagrama con los procesos creados. Identifica cada proceso con una letra e indica lo que imprimirán por la terminal cada uno de los procesos.

```
main(){
    int a[2];

    a[0] = a [1] = 0;
    for ( i = 0; i < 3; ++i){
        if (fork()) ++a[0];
        else ++a[1];
    }
    printf ("(%d,%d)",a[0],a[1]);
}
```

   Indica qué problema de diseño tiene el código presentado. Procesos huérfanos, procesos zombies, etc.
6. Se realiza el código adjunto. Realiza un diagrama con los procesos creados.

```
int main(){
    if( !fork()){
        fork();
        wait();
    } else {
        fork();
        fork();
        wait();
    }
    exit(EXIT_SUCCESS);
}
```

   Se pide:

   A. Dibujar la jerarquía de procesos que resulta de la ejecución de dicho código. Identifica cada proceso con una letra o numeración distinta. ¿Cuántos procesos, contando el proceso padre, se habrán generado en la ejecución del código?

   B. Indica qué problema de diseño tiene el código presentado. Procesos huérfanos, procesos zombies, etc ...
7. Se realiza el código adjunto. Realiza un diagrama con los procesos creados. Identifica cada proceso con una letra e indica lo que imprimirán por la terminal cada uno de los procesos.

```
int main(){
    int i, pid;

    for ( i = 0; i < 3; i++){
        pid=fork();
        if(!i%2) fork();
    }

    for( i = 0; i < 5 ; ++i) wait();
    exit(EXIT_SUCCESS);
}
```

   Se pide:

A. Dibujar la jerarquía de procesos que resulta de la ejecución de dicho código. Identifica cada proceso con una letra o numeración distinta. ¿Cuántos procesos, contando el proceso padre, se habrán generado en la ejecución del código?

B. Indica qué problema de diseño tiene el código presentado. Procesos huérfanos, procesos zombies, etc ...

8. Se realiza el código adjunto. Realiza un diagrama con los procesos creados. Identifica cada proceso con una letra e indica lo que imprimirán por la terminal cada uno de los procesos.

```
main(){
      if(fork()){
          wait();
          if(fork()) wait();
      } else {
          fork();
          wait();
      }
      exit(0);
  }
```

Se pide:

A. Dibujar la jerarquía de procesos que resulta de la ejecución de dicho código. Identifica cada proceso con una letra o numeración distinta. ¿Cuántos procesos, contando el proceso padre, se habrán generado en la ejecución del código?

B. Indica qué problema de diseño tiene el código presentado. Procesos huérfanos, procesos zombies, etc ...

9. Se realiza el código adjunto.

```
main(){
    int i=0;
    pid_t pid;

    while((pid = fork()) != 0 && i < 1){
        if(pid) fork();
        else wait();
        ++i;
    }
    wait();
}
```

Se pide:

A. Dibuja la jerarquía de procesos que resulta de la ejecución de dicho código. Identifica cada proceso con una letra o numeración distinta. ¿Cuántos procesos, contando el proceso padre, se habrán generado en la ejecución del código?

B. Indica qué problema de diseño tiene el código presentado. Procesos huérfanos, procesos zombies, etc ...