Computing Collatz Sequence in a forked process (POSIX IPC Assignment)

Overview:

The aim of the assignment is to implement IPC through *POSIX Shared Memory*. The system calls used are `shm_open()` to open the shared memory, `ftruncate()` to allocate space to the shared memory segment, and `shm_unlink()` to close the shared memory segment.

Explanation:

Before forking:

Before forking, all the variables are declared.
A shared memory segment is opened using `shm_open()` and it's file descriptor is stored in `int memd`.
`memd = shm_open("/shared_object", O_CREAT+O_RDWR, 0666);`
The shared memory segment is established before forking so that its descriptor can be used by both processes after forking.

Forking:

The process is forked, and a `switch...case` conditional is employed to differentiate between the child and the parent.

```
switch(pid = fork()) {
    case -1:
        // Error forking

    case 0:
        // Child process

    default:
        // Parent process
}
```

Child process:

The child process accepts a number and verifies if it's greater than 1. It then computes the collatz sequence and stores it in a dynamically sized integer array.

```
scanf("%d", &x);
if(x < 1){
    printf("ERROR: Enter a number greater than 1.\n");
    exit(0);
}
n = (int*) malloc(sizeof(int));

for (i = 0; x != 1; i++){
    // Compute Collatz Sequence, store in n[]
}
```

The shared memory segment is then resized to have the same size as the array, and the array is copied into the shared memory segment using `memcpy` from the string.h library.

```
ftruncate(memd, i*sizeof(int));
memcpy(mmap(NULL, i*sizeof(int), PROT_READ | PROT_WRITE, MAP_SHARED, memd, 0), n,
 (i+1)*sizeof(int));
```

After this, the child process exits with a ***return value equal to the size of the array.***
`exit(i);`
This return value is used in the parent process so that we can read the exact amount of memory as is necessary.

Parent Process:

The parent process waits for the child process to end, and stores return status in `int prv`. From this, the exact return value (which is the size of the array stored in the shared memory segment) is extracted and stored in `prv`.

```
wait(&prv);
prv = WEXITSTATUS(prv);
```

It checks if the child executed succesfully, and if it did, it reads the shared memory segment using `mmap()` and stores its contents in an integer array `n[]`. The contents of `n[]` are then printed.

```
n = mmap(NULL, prv*sizeof(int), PROT_READ | PROT_WRITE, MAP_SHARED, memd, 0);
for (i = 0; i <= prv; i++){
    printf("%d\t", n[i]);
}
printf("\n");
```

The shared memory segment is unlinked before the parent process exits.
`shm_unlink("/shared_object");`

Output:

When given a number, the program prints the Collatz Sequence.

```
$ ./a.out
8
8    4    2    1
```

If the number is less than 1, it throws an error an quits to the shell with an exit code 1.

```
$ ./a.out
-3
ERROR: Enter a number greater than 1.
```