Lab6 Report

The following report contains the steps involved in changing the forking process to **Copy on Write.**
PART 1:

1. First we need to change UVMCOPY which directly allocates new memory when forking is done so instead of that we can just allocate the parent's pages to the child process.
2. For this we first take one of 8th or 9th bit for what is known as the COW BIT.This tells whether a page is a normal page or copy on write page.
3. we define PTE_C to define the cow bit which is defined as **1L<<8**.
4. Now in the UVMCOPY function we change the pte value at the PTE_C to 1 and PTE_W to 0 so it is not writable.

Now onto the second part of the assignment.
PART 2:

1. we get the error due to write in **trap.c**,so we need to identify the errors due to write,which can be done using the r_scause()==15 which means it is error due to write.
2. now we need to differentiate between the genuine write errors and the errors due to copy on write.
3. For that we check whether the cow bit is 1 or not ,if it is 1 then the write error is caused to the **COW** bit so we need to act appropriately else it is just another write error,so we just print the errors as usual.
4. For the **COW** part since it is a write error now we need to allocate a new page and remove the mapping to the old page and allocate them to the new page .
   - the first part is done using UVMUNMAP and the second part using MAPPAGES.
   - during mapping the flag bits are changed appropriately.

Now that we are done with the second part we have to go to the third part which is
PART 3:

1. Here we want to avoid the fact that we cannot tell whether a page has only one mapping or not when it has the last mapping then when we allocate a new page then we need to free the old page
2. In order to know this we have an array which stores that number of mappings for each page now we know the mappings we increase when we are in uvmcopy and initialize to 1 when we do kalloc(); and decrease when ever we go to kfree();
3. We are modifying kfree such that it only frees the page when the mappings are zero .
4. In this way we complete the COW allocation.

Onto the last Part

PART 4:

1. We also modify copy out similar to the modifications done in user trap
2. Refer to the process in kernel/vm.c/copyout(...) for the process.