**Name – Arighna Chakraborty Roll – 511**

**Answers**

**Question 1.**

#include <stdio.h>

#include <stlib.h>

#include<unistd.h>

#include<fcntl.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<string.h>

int main(int argc, char \*argv){

int f;

char \*buffer;

char \*filename = arg[1];

if(argc != 2){

printf(“Invalid number of arguments”);

exit(1);

}

f = open(filename, O\_RDWR | O\_TRUNC, 0777);

if( f == -1){

printf(“Could not open the file”);

exit(1);

}

while(read(filename, buffer, 1) > 0){

if(buffer == ‘a’ || buffer == ‘A’)

write(filename, ‘$’, 1);

}

int offset = lseek(filename, 0, SEEK\_SET);

printf(“Modified file is :\n”);

while(read(filename, buffer, 1) > 0){

printf(“%s”, buffer);

}

return 0;

}

**Question 2.**

#include <stdio.h>

#include <stlib.h>

#include<unistd.h>

#include<fcntl.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<string.h>

int main(){

int curr\_id, child\_id;

int status;

char buffer[1024];

curr\_id = getpid();

printf(“pid : %d – Parent Process about to call fork\n”, curr\_id);

child\_id = fork();

if(child\_id != 0){

curr\_id = getpid();

printf(“pid : %d – Just forked a child process with id %d\n”, curr\_id, child\_id);

printf(“pid : %d – Waiting for the child process to complete\n”);

wait(&status);

printf(“pid : %d – Child process has completed with status : %d\n”, curr\_id, status);

}

else {

curr\_id = getpid();

prinf(“pid : %d – Child process to print current working directory\n”, curr\_id);

getcwd(buffer, sizeof(buffer));

printf(“%s\n”, buffer);

}

printf(“pid : %d Exiting\n”, curr\_id);

exit(0);

}