Q 1.) Write a C Program to Create a Process Using fork() command. Show the Id's of the Parent and Child Process

```
#include <stdio.h>


int main(){

   int cid = fork();

   if(cid==0){

      printf("Child process created with id : %d\n",getpid());

      printf("Process id of parent process : %d\n",getppid());

   }

}
```

Output:

```
anand@DESKTOP-NK1DJIT MINGW64 /e/sem5labs/os_lab/lab 2 (master)
$ gcc 1.c

anand@DESKTOP-NK1DJIT MINGW64 /e/sem5labs/os_lab/lab 2 (master)
$ ./a.exe
Child process created with id : 1001
Process id of parent process : 1000
```

Q 2.) 2. Write a C program To create child with sleep command using getpid.

STEP 1: Start the execution and create a process using fork() command.

STEP 2: Make the parent process to sleep for 10 seconds.

STEP 3:In the child process print it pid and it corresponding pid.

STEP 4: Make the child process to sleep for 5 seconds.

STEP 5: Again print it pid and it parent pid.

STEP 6: After making the sleep for the parent process for 10 seconds print it pid.

STEP 7: Stop the execution.

```
 #include <stdio.h>
int main(){
   int pid = fork();
   if(pid == -1){
      printf("Child process could not be started\n");
   }else if(pid == 0){
      printf("Child process started with id :%d\n",getpid());
      printf("Parent porcess id : %d\n",getppid());
      printf("Child is going to sleep\n");
      sleep(5);
      printf("Child is back\n");
      printf("Child process id :%d\n",getpid());
      printf("Parent process id :%d\n",getppid());
   }else{
      printf("Parent is going to sleep\n");
      sleep(10);
      printf("Parent is back\n");
      printf("Parent process id %d\n",getpid());
   }
}
```

Output:

```
anand@DESKTOP-NK1DJIT MINGW64 /e/sem5labs/os_lab/lab 2 (master)
$ gcc 2.c

anand@DESKTOP-NK1DJIT MINGW64 /e/sem5labs/os_lab/lab 2 (master)
$ ./a.exe
Child process started with id :1001Parent is going to sleep

Parent porcess id : 1000
Child is going to sleep
Child is back
Child process id :1001
Parent process id :1000
Parent is back
Parent process id 1000
```

Q 3.)Perform wait command using c program.

STEP 1:Start the execution

STEP 2:Create process using fork and assign it to a variable

STEP 3:Check for the condition pid is equal to 0

STEP 4:If it is true print the value of i and teriminate the child process

STEP 5:If it is not a parent process has to wait until the child teriminate

STEP 6:Stop the execution

```
#include <stdio.h>

#include <sys/wait.h>

#include<sys/types.h>

#include<unistd.h>

int main(){

   int pid = fork();

   if(pid == -1){

      printf("Child process could not be started\n");

   }else if(pid == 0){
```

```
        printf("Child process started with id :%d\n",getpid());
        printf("Parent porcess id : %d\n",getppid());
        printf("Child is going to sleep\n");
        sleep(2);
    }else{
        printf("Parent wait started\n");
        wait(NULL);
        printf("Parent wait ended\n");
    }
}
```

Output:

```
devesh2997@DESKTOP-NK1DJIT:~/sem5labs/os_lab/lab 2$ gcc 3.c
devesh2997@DESKTOP-NK1DJIT:~/sem5labs/os_lab/lab 2$ ./a.out
Parent wait started
Child process started with id :123
Parent porcess id : 122
Child is going to sleep
Parent wait ended
```

Q 4.)4. Write a C program :

(a) To create a file

(b) To write in that file

(c) To read the ultimate file

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
```

```
#include <fcntl.h>

#include <string.h>


int main(){

   FILE  *i =fopen("foo.txt","w+");

   char text[] = "hello world\n";

   fputs(text, i);

   rewind(i);


   char str1[100], str2[100];

   fscanf(i, "%s %s",str1,str2);

   printf("Contents of file: %s %s\n",str1,str2);

}
```

Output:

```
devesh2997@DESKTOP-NK1DJIT:~/sem5labs/os_lab/lab 2$ gcc 4.c
devesh2997@DESKTOP-NK1DJIT:~/sem5labs/os_lab/lab 2$ ./a.out
Contents of file: hello world
```

Q 5.)Write a C program for following situation:

Take your own name as an input and print that character-wise after waiting specific time interval.Time interval for the next character = index of that character

Example : Input : "SAUMYA"

```
#include <stdio.h>

#include <string.h>

#include<unistd.h>

int main(){
```

```
    char name[100];
    scanf("%s",name);
    int n = strlen(name);
    int i;
    for(i=0;i<n;i++){
        sleep(i);
        printf("%c",name[i]);
        fflush(stdout);
    }
    printf("\n");
}
```

Output: