**Assignment 2**

**Sriram R SRN : PES1UG20CS435 Section : H**

**Code :**

#include <stdio.h>

#include <string.h>

#include <unistd.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/wait.h>

#define ARRAY\_SIZE 8

#define BUFFER\_SIZE 8 // should be greater than expected number of digits in sum

#define READ\_END 0

#define WRITE\_END 1

int main()

{

int numArray[] = {1, 2, 3, 4, 5, 6, 7, 8};

int sumChild = 0;

int sum = 0;

char write\_msg[BUFFER\_SIZE] = "";

char read\_msg[BUFFER\_SIZE] = "";

int fd[2];

pid\_t pid;

if (pipe(fd) == -1)

{

printf("pipe failure\n");

return 1;

}

pid = fork();

if (pid > 0) // parent process - calculates sum of first half and stores in pipe

{

for(int i = 0; i < (ARRAY\_SIZE / 2); i++) sum += numArray[i];

snprintf(write\_msg, BUFFER\_SIZE, "%d", sum); // converts integer to string (itoa not supported)

printf(".... parent process is storing %s in pipe ....\n", write\_msg);

close(fd[READ\_END]);

write(fd[WRITE\_END], write\_msg, strlen(write\_msg) + 1);

close(fd[WRITE\_END]);

wait(NULL);

}

else if (pid == 0) // child process - calculates sum of second half, adds to the sum in the pipe and prints

{

close(fd[WRITE\_END]);

read(fd[READ\_END], read\_msg, BUFFER\_SIZE);

for(int i = (ARRAY\_SIZE / 2); i < ARRAY\_SIZE; i++) sumChild += numArray[i];

sumChild += atoi(read\_msg);

printf("child process summation : %d\n", sumChild);

close(fd[READ\_END]);

}

else

{

printf("fork failure\n");

return 1;

}

return 0;

}

**Output (window highlighted in red) :**

