```
#include <iostream>
#include <sys/wait.h>
#include <sys/types.h>
#include <unistd.h>
using namespace std;


int sumOfArr(int *arr){
        int sum = 0;
        for(int i = 0; arr[i]!=-9999; i++){
                sum +=arr[i];
        }
        return sum;
}
int size(int *arr){
        int Size = 0;
        for(int i = 0; arr[i]!=-9999; i++){
                Size++;
        }
        return Size-1;
}

void average(int *arr){
        cout<<"\nAverage : "<<sumOfArr(arr)/size(arr);
}

void maximumNumber(int *arr){
        int max = -999999999;
        for(int i = 0 ; arr[i] != -9999; i++){
                if(max<arr[i]){
                        max = arr[i];
                }
        }
        cout<<"\nMax size in array : "<<max;
}


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

        int *arr = new int[argc];
        for (int i = 1; i < argc; ++i){
                int index = 0;
                int number = 0;

                while(argv[i][index] !='\0'){
                        number *= 10;
                        number += argv[i][index]-48;
                        index++;
                }
                arr[i-1] = number;
        }
        arr[argc] = -9999; // I'm considering that array terminate
with -9999

        pid_t fFID;// first Fork ID
        fFID = fork(); // creat child and parent processes
```

```
        if(fFID==0){ // first child process
                cout<<"\nEnter in first child process";
                cout<<"\nsum of all element of array is : "<<
sumOfArr(arr)<<endl;
        }

        else if(fFID>0){ // PARENT PROCESS
                pid_t fSID = fork();

                if(fSID == 0){ // second child
                        cout<<"\nEnter in second child process";
                        average(arr);
                }
                else if(fSID>0) {// parent process
                        wait(NULL);
                        pid_t fTID = fork(); // third child;
                        if(fTID == 0 ){ // third child
                                cout<<"\nEnter in third child
process";
                                maximumNumber(arr);
                        }
                }
        }
        return 0;
}
```