

SUM & MEAN

simple terminal app example in c language

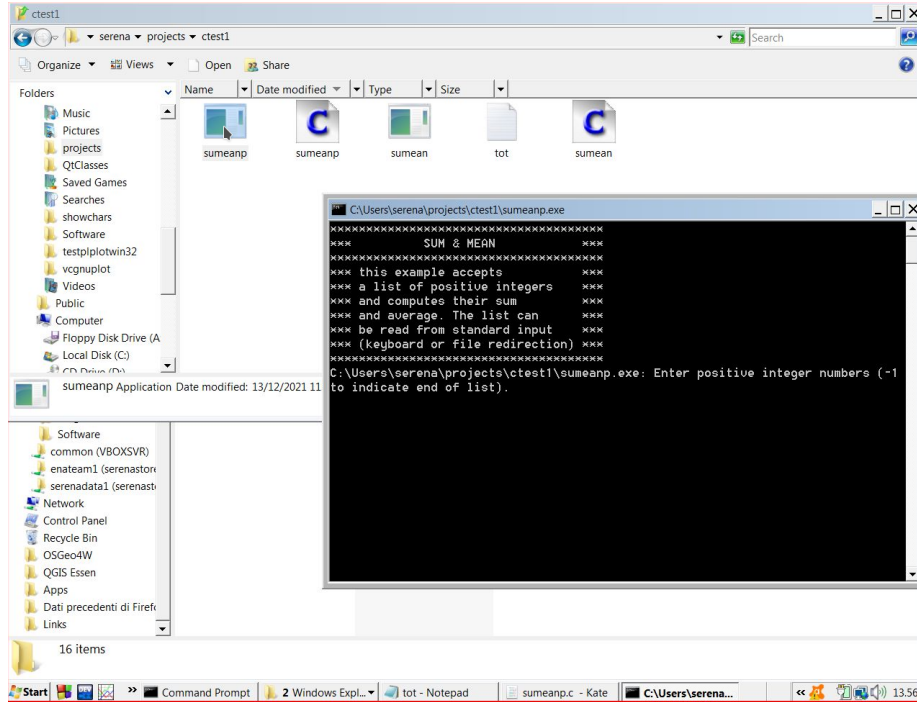


Figure 1: sum & mean execution

Introduction

This programming example accepts in input a list of positive integers and computes their sum and average, showing them as output. At the end of the execution it shows the number m of input 1-D points, their sum

$$\sum_{n=1}^m (p_n)$$

Figure 2: sum formula

and their average value

$$\frac{\sum_{n=1}^m (p_n)}{m}$$

Figure 3: mean formula

The list can be read from standard input (keyboard or file redirection).

The log of what's going on with the computation is printed on the standard error output stream, while the actual numerical output is printed on the standard output at the end. This allows to use the app output as input of a following app, that read from the standard input. The output created on the standard input and/or the output created on the standard error can be also saved in different text files.

How To

The following instructions are given supposing you're yousing the GNU gcc c language compiler. You need to use a terminal emulator, like xterm on MacOS or gnome-terminal or konsole on GNU-Linux or cmd terminal on Microsoft Windows. Not all the instructions are given for all different operating systems but understanding the main examples, they may be derived easily

to compile on MSwin mingw32:

```
mingw32-make -f Makefile.win sumean
```

to compile on linux/MacOS/MS-Win+msys2:

```
make sumean
```

to run on MS-Win cmd shell

```
sumean
```

to remove all created files in MS-Win cmd shell

```
del sumean.o sumean *.log *.asc
```

to run on linux/MacOS/MS-Win+msys2

```
./sumean
```

to run on linux/MacOS/MS-Win+msys2 saving the results on an output file

```
./sumean >out.asc
```

to run on linux/MacOS/MS-Win+msys2 & logging

saving the instructions on an output file err.log saving the results on an output file out.asc this implies you giving input without hints (blind mode)

```
./sumean 2>err.log >out.asc
```

to remove all created files in GNU-linux/MacOS/MS-win+msys2

```
rm sumean.o sumean *.log *.asc
```

Using file explorer and mouse

the source file sumeanp.c is just the same app code with added a feature useful to run the app from file explorer graphical user interface. The letter p at the end of the file name means pause. After compiling you can just click on the sumeanp.exe. It is named sumeanp.exe but take in mind that using Microsoft Windows file explorer the extension is written in a different column in the details view, or you can distinguish the file type from the different icon. Just compile the source.

to compile on GNU-Linux/MacOS:

```
make sumeanp
```

to compile on MS-Win:

```
mingw32-make -f Makefile.win sumeanp
```

```
sumean : bash — Konsole
File Edit View Bookmarks Settings Help
francesco@squirrel:~/Documents/Miscellanea/sumean$ ./sumean
*****
***          SUM & MEAN          ***
*****
*** this example accepts          ***
*** a list of positive integers   ***
*** and computes their sum        ***
*** and average. The list can     ***
*** be read from standard input   ***
*** (keyboard or file redirection) ***
*****
./sumean: Enter positive integer numbers (-1 to indicate end of list).
10
12
2
4
-1
./sumean: you got 0 warnings
./sumean: valid numbers read: 4
./sumean: sum: 28.000000
./sumean: average: 7.000000
./sumean: showing results:
#points sum          average
4      28.000000      7.000000
francesco@squirrel:~/Documents/Miscellanea/sumean$
```

Figure 4: sum & mean results

Installation

The sunmean app do not really need to be installed permanently on the system to be executed and exercise with it, it can be downloaded and saved in any folder of your computer, then compiled and run. Of course you need the permission to write on that folder, having it in a subfolder of the Desktop is ok. Some antiviruses and security systems on some OSs may wrongly detect the generated exe as harmful, take care of instruct or temporary stop/disable that systems. Many modern GNU-Linux distribution do not allow as default to run a binary exe app clicking on it with the mouse by the file browser. The Gnome file browser nautilus (now called only Files) can be instructed to do it, but the safer way to run it from Graphical User Interface (GUI) on GNU-Linux is to use a .desktop file as launcher and clicking on it instead. The file install.sh and uninstall.sh are there for that purpose, for people that wanna try this optional method.

Troubleshooting

In the simpler case, we're supposing that the gcc compiler executable can be called with the following instruction: `gcc sumean.c -o sumean`, which implies that somewhere in the list of the PATH environment variable is included a path to a folder containing the gcc executable file or a link to it. In the simpler case in *nix like OSs it is called just gcc and in Microsoft Windows it should be called gcc.exe .

```
C:\Users\francesco\Desktop\sumeant>echo %PATH%
C:\Python27\;C:\Python27\Scripts;C:\Python34\;C:\Python34\Scripts;C:\windows\system32;C:\w
ndows;C:\windows\system32\wbem;C:\windows\system32\WindowsPowerShell\v1.0;C:\Program Fil
"\"Microsoft Visual Studio 9.0\VC\bin\;C:\Dev-Cpp\;C:\Make\bin\;C:\Program Files\Microsof
indows Performance Toolkit\;C:\win32app\FTN95\;C:\Program Files\Microsoft Visual Studio 9
\VC\bin\amd64\";C:\Program Files\Microsoft Visual Studio 9.0\VC\bin\amd64\;C:\win32app\
lford\;C:\Dev-Cpp\;C:\Program Files\Microsoft Visual Studio 9.0\VC\;C:\Program Files\
icrosoft Visual Studio 9.0\VC\bin\;C:\Dev-Cpp\bin\;C:\gnuplot\bin

C:\Users\francesco\Desktop\sumeant>dir C:\Dev-Cpp\bin\gcc.exe
Volume in drive C has no label.
Volume Serial Number is 0000-0000

Directory of C:\Dev-Cpp\bin

29/01/2005    23:35           88,064  gcc.exe
             1 file                88,064 bytes
             0 directories        8,382,214,144 bytes free

C:\Users\francesco\Desktop\sumeant>_
```

Figure 5: path to gcc on MS-win OS

Modern mingw32-gcc releases have that filename called

x86_64-w64-mingw32-gcc.exe for the 64 bit version and i686-w64-mingw32-gcc.exe for the 32 bit version, but may also include a copy called gcc.exe as well.

Computing Steps

The following are the processing steps executed by the program.

- Present to the user what the app is doing
- Tell user to enter data points; maximum of allowed pts = 39
- Start an infinite loop, flow interrupted only by break;
 - Read a value; end of file (EOF val = -1) will be checked
 - Value input before is in string format, if the string correctly express a legal numerical value, convert it into integer
 - Insert the read item in the 1-D points array and increment the partial sum with its value
 - Reject the item value otherwise
 - check when the end of the dataset is reached (reading -1)
 - if -1 read \rightarrow break; continue the loop otherwise
- EOF detected : compute average
- print final report #points, sum, average.

TODOs

Execution examples with remarkable datasets can be added. Example of how to run it and catch the app output from another app reading from stdin will be useful to show how to create a simple software pipeline. Another useful example to add will be using an external app that read the sumean output and graph the points succession and the mean value, and the sum.

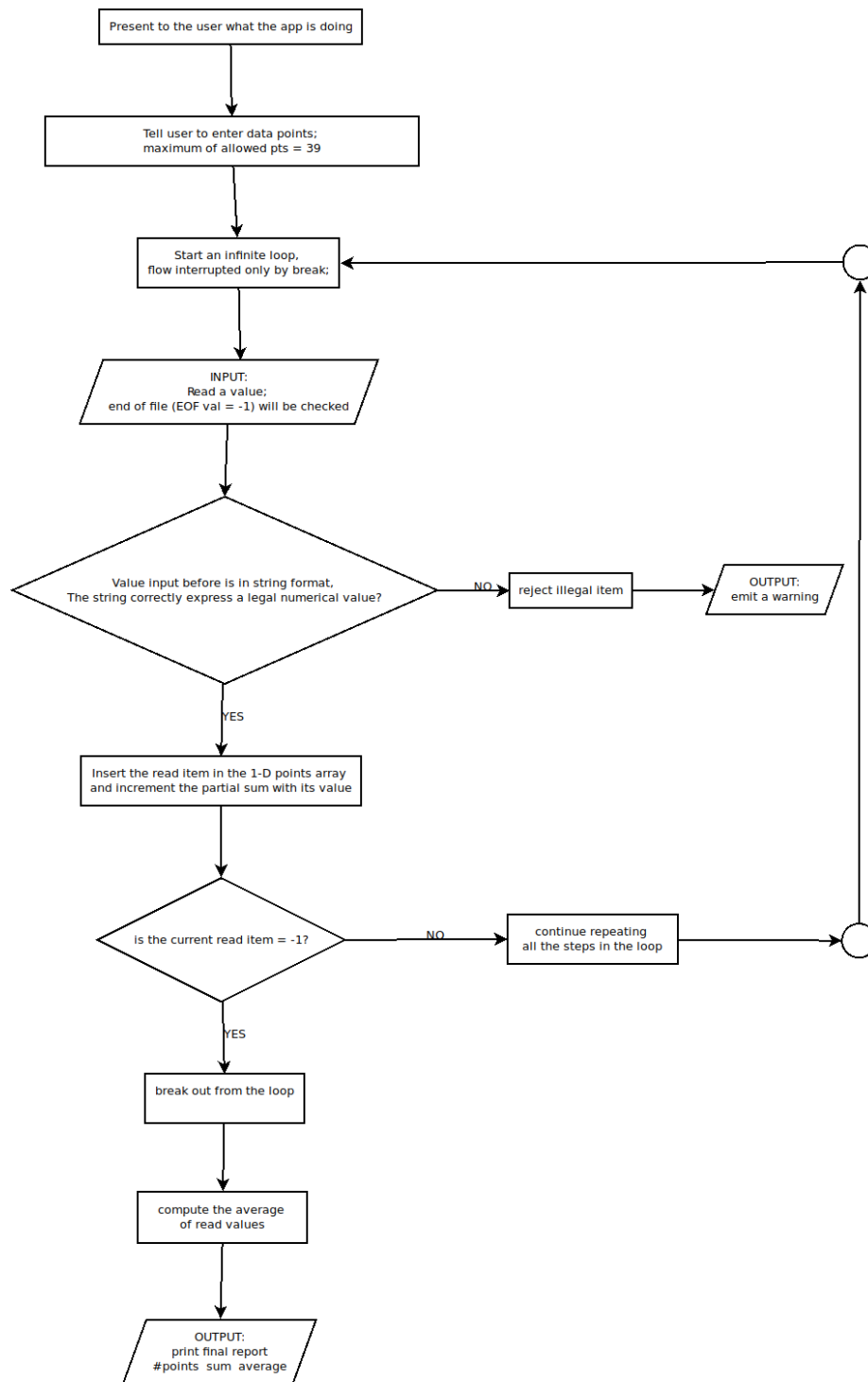


Figure 6: flow-chart of SUM & MEAN