```
it@ITs-MacBook-Air 30seconds % time ./a.out
./a.out  30.53s user 0.07s system 99% cpu 30.734 total
it@ITs-MacBook-Air 75seconds % time ./a.out
./a.out  72.55s user 0.39s system 99% cpu 1:13.08 total
it@ITs-MacBook-Air 120seconds % time ./a.out
./a.out  121.58s user 0.57s system 99% cpu 2:02.50 total
```

```
it@ITs-MacBook-Air 30seconds % bash script.sh
 30seconds main Program  Running

real    0m30.749s
user    0m30.544s
sys     0m0.097s
it@ITs-MacBook-Air 75seconds % bash script.sh
75seconds main Program  Running

real    1m12.696s
user    1m12.360s
sys     0m0.211s
it@ITs-MacBook-Air 120seconds % bash script.sh
 120seconds main Program  Running

real    2m2.344s
user    2m1.637s
sys     0m0.452s
```

For Background:

```
it@ITs-MacBook-Air 30seconds % time ./a.out &
[1] 28462
it@ITs-MacBook-Air 30seconds % ps
it@ITs-MacBook-Air 30seconds % ./a.out  30.57s user 0.14s system 99% cpu 30.718 tota
it@ITs-MacBook-Air 75seconds % time ./a.out &
[1] 23166
it@ITs-MacBook-Air 75seconds % ps
it@ITs-MacBook-Air 75seconds % ./a.out  72.46s user 0.28s system 99% cpu 1:12.76 tot
it@ITs-MacBook-Air 120seconds % time ./a.out &
[2] 23299
it@ITs-MacBook-Air 120seconds % ps
it@ITs-MacBook-Air 120seconds % ./a.out  122.25s user 0.73s system 99% cpu 2:03.47 t
```

Shell

```
#!/bin/sh
#Vitaliy Prymak
PROG1="main";
# 30 Second Program
# g++ compilation
success=$(g++ $PROG1.cpp -o  $PROG1)
# Check if compilation not fails
if [ $success == ""]
then
    echo " 30seconds $PROG1 Program  Running";    # echo output
    time $(./$PROG1);          # timing the program

else
    echo $success;
fi
```

1) What was the difference in run time for running in foreground or background Be detailed?

Generally Runtime of foreground and background process are similar. But Background Process has more resources hence it works faster.

Foreground : 30.74s 72.69s 122.34s

Background: 30.57s 72.46s 122.25s

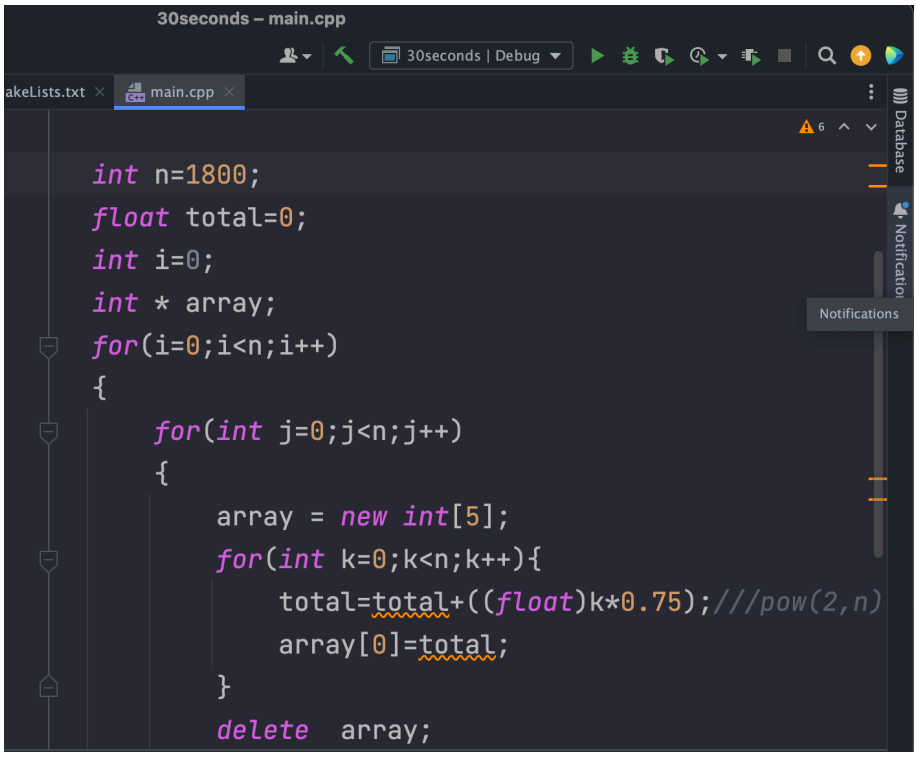
2) What is the difference between a foreground and a background process?

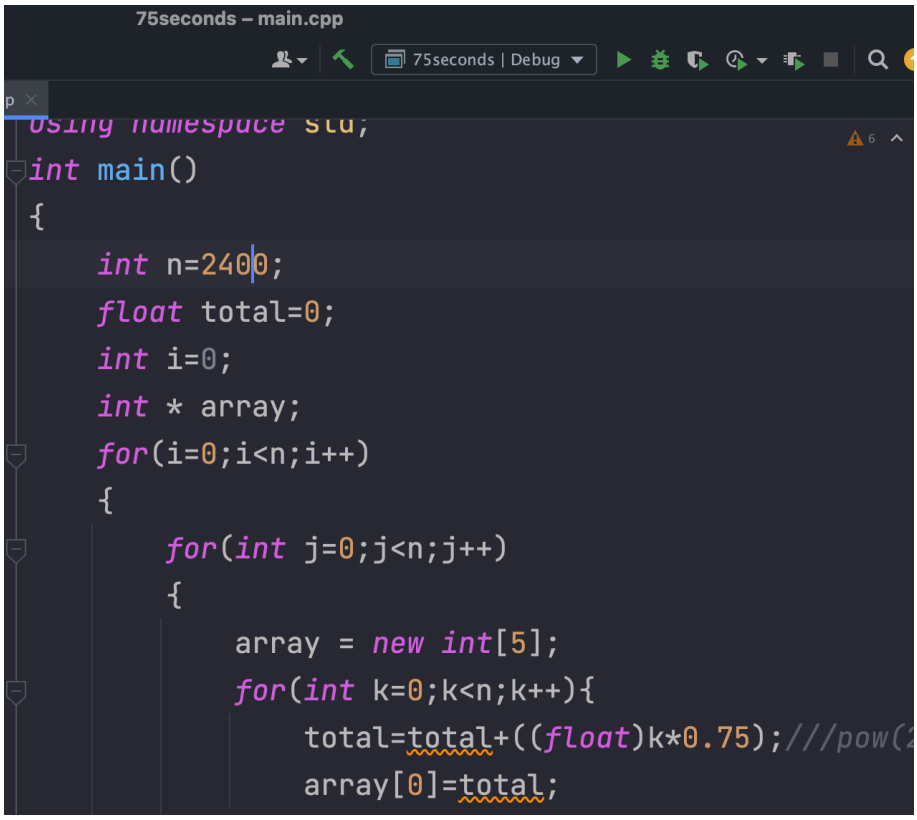
Foreground Process: Runs on foreground, directly visible on main UI. a job that runs in the forefront and wait for it to finish is referred to as a foreground process.

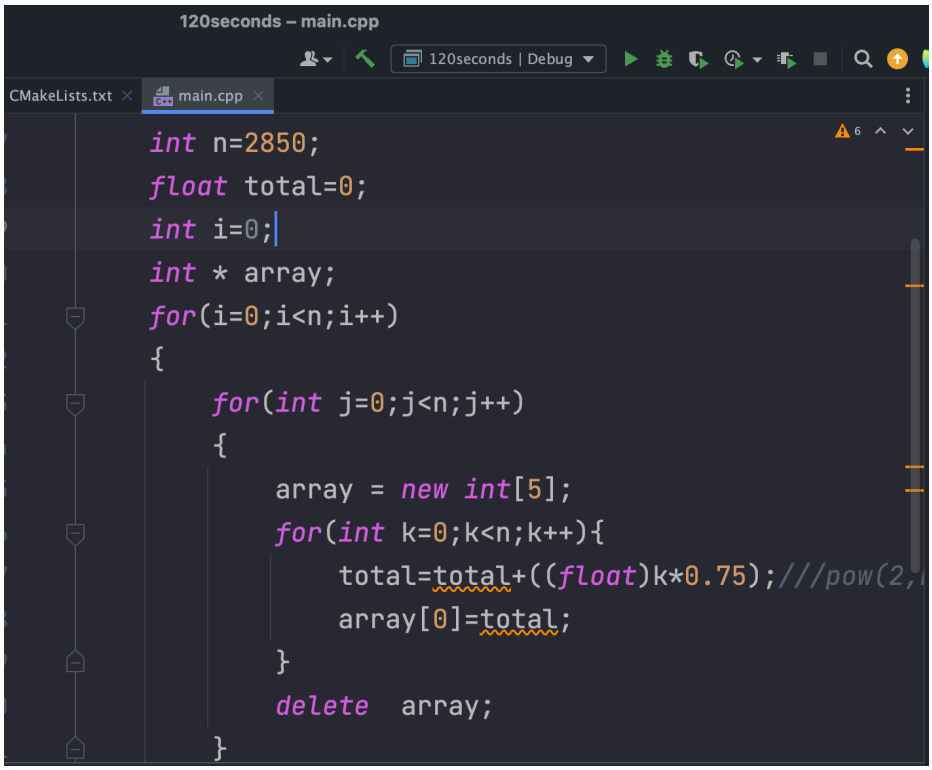
Background Process: Runs on background, not visible on main UI. Can be seen running in process table. In this the shell does not have to wait for a background process to complete before running additional. Hence they are faster.

3) Why is there a run time difference between foreground and background process that is running the same program?

Shell does not have to wait for a background process to complete before running additional. Hence background are faster.


30seconds – main.cpp
30seconds | Debug
akeLists.txt
main.cpp
Database
Notifications
int n=1800;
float total=0;
int i=0;
int * array;
for(i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
array = new int[5];
for(int k=0;k<n;k++){
total=total+((float)k*0.75);///pow(2,n)
array[0]=total;
}
delete array;


75seconds – main.cpp
75seconds | Debug
using namespace std;
int main()
{
int n=2400;
float total=0;
int i=0;
int * array;
for(i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
array = new int[5];
for(int k=0;k<n;k++){
total=total+((float)k*0.75);///pow(
array[0]=total;


120seconds – main.cpp
120seconds | Debug
CMakeLists.txt
main.cpp
int n=2850;
float total=0;
int i=0;
int * array;
for(i=0;i<n;i++)
{
for(int j=0;j<n;j++)
{
array = new int[5];
for(int k=0;k<n;k++){
total=total+((float)k*0.75);///pow(2,
array[0]=total;
}
delete array;
}