Cel:

- Przeprowadzenie pomiaru czasu CPU i zegarowego wykonania operacji
- Organizacja środowiska tworzenia oprogramowania w systemie Linux (make, cc itp.)

Wykonanie:

1. Utworzenie katalogów, pobranie plików ze strony przedmiotu i rozpakowanie pliku pomiar_czasu.tgz.
   ```~/$ mkdir -p PR_lab/lab_1 PR_lab/pomiar_czasu
   ~/$ cd PR_lab
   ~/PR_lab$ wget ww1.metal.agh.edu.pl/~banas/PR/pomiar_czasu.tgz
   ~/PR_lab$ tar xvzf pomiar_czasu.tgz
   ~/PR_lab$ mv pomiar_czasu.* pomiar_czasu/
   ~/PR_lab$ mv Makefile lab_1/
   ~/PR_lab$ cd lab_1/
   ~/PR_lab/lab_1$ wget ww1.metal.agh.edu.pl/~banas/PR/moj_program.c```

2. Edycja pobranego programu mój_program.c

   Dodanie pliku nagłówkowego pomiar_czasu.h
   #include „pomiar_czasu.h”

   Dodanie dyrektywy POMIAR_CZASU
   #define POMIAR_CZASU

   Dodanie int przed funkcją main

```c
#include "pomiar_czasu.h"
#define POMIAR_CZASU

const int liczba = 100000;

int main(){
```

   Dodanie dwóch rodzajów pomiarów czasu

```c
#ifdef POMIAR_CZASU
inicjuj_czas();
#endif
for(i=0;i<liczba;i++){

  printf("%d ",k+i);

}
#ifdef POMIAR_CZASU
drukuj_czas();
#endif
```

```c
#ifdef POMIAR_CZASU
double t1=czas_zegara();
double t2=czas_CPU();
#endif
for(i=0;i<liczba;i++){

  a = 1.000001*a+0.000001;

}
#ifdef POMIAR_CZASU
t1=czas_zegara()-t1;
t2=czas_CPU()-t2;

printf("czas CPU: %lf\n",t1);
printf("czas zegarowy: %lf\n", t2);
#endif
```
```

3. Modyfikacja pliku Makefile

Odkomentowanie linijki z plikiem nagłówkowym
Odkomentowanie i edycja linijki z bibliotekami
Edycja kompilacji pliku moj_program.o
Dodanie kompilacji pliku mój_program.c

```
# pliki naglowkowe
INC = -I../pomiar_czasu

# biblioteki
LIB = -L../pomiar_czasu -lm -lpomiar_czasu

# zaleznosci i komendy
moj_program: moj_program.o
    $(LOADER) $(OPT) moj_program.o -o moj_program $(LIB)

# jak uzyskac plik moj_program.o ?
moj_program.o: moj_program.c
    $(CCOMP) -c $(OPT) $(INC) moj_program.c

clean:
    rm -f *.o
```

4. Stworzenie pliku pomiar_czasu.o
   *~/PR_lab/pomiar_czasu$ gcc -c pomiar_czasu.c*

5. Utworzenie w katalogu „pomiar_czasu" statycznej biblioteki z procedurami pomiaru czasu
   *~/PR_lab/pomiar_czasu$ ar -rs libpomiar_czasu.a pomiar_czasu.o*

6. Kompilacja w folderze lab_1 dla wersji do debugowania oraz wersji zoptymalizowanej ze stukrotnie zwiększoną liczbą pętli z operacją arytmetyczną.

```
const int liczba = 10000000;
```

Zmiana wartości w pliku mój_pomiar.c

```
litewka_jakub@honorata:~/PR_lab/lab_1$ make
gcc -c -g -DDEBUG -I../pomiar_czasu moj_program.c
gcc -g -DDEBUG moj_program.o -o moj_program -L../pomiar_czasu -lm
lpomiar_czasu
```

Uruchomienie programu poleceniem ./moj_porgram

```
49999 czas standardowy = 2.854858
czas CPU         = 2.192625
czas zegarowy    = 38.967227

Czas wykonania 10000000 operacji wejscia/wyjscia:
czas CPU: 0.081126
czas zegarowy: 0.081124
Wynik operacji arytmetycznych: 44051.733319
Czas wykonania 10000000 operacji arytmetycznych:
litewka_jakub@honorata:~/PR_lab/lab_1$
```

Wyniki pomiarów otrzymanych za pomocą programu

```
49999 czas standardowy = 2.937659
czas CPU         = 2.181171
czas zegarowy    = 41.651027

Czas wykonania 10000000 operacji wejscia/wyjscia:
czas zegarowy: 0.048604
czas CPU: 0.048331
Wynik operacji arytmetycznych: 44051.733319
Czas wykonania 10000000 operacji arytmetycznych:
litewka_jakub@honorata:~/PR_lab/lab_1$
```

Struktura plików

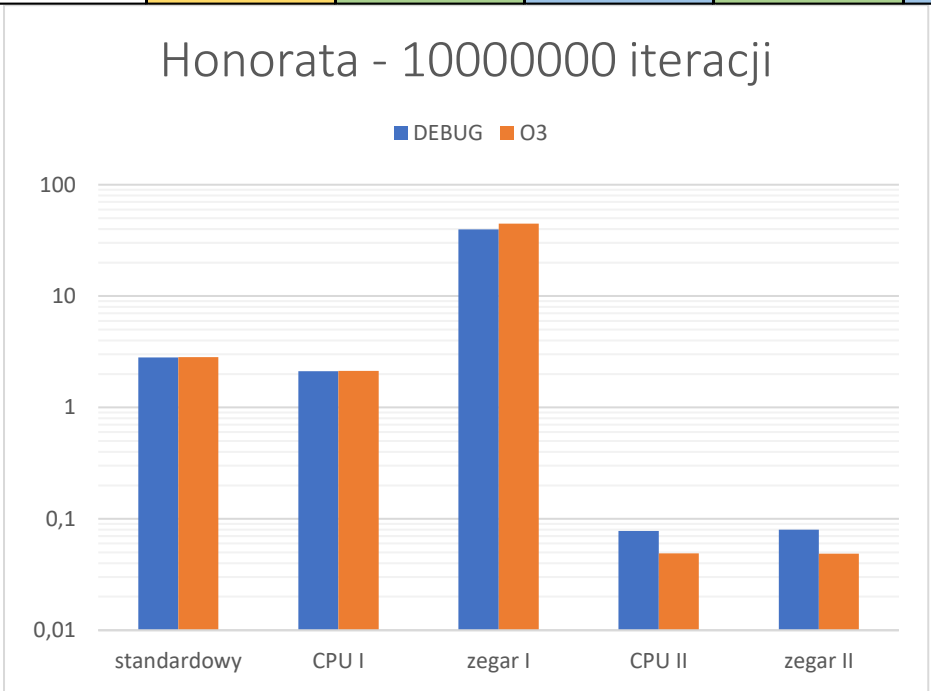
```
.
└── PR_lab
    ├── lab_1
    │   ├── Makefile
    │   ├── moj_program
    │   ├── moj_program.c
    │   └── moj_program.o
    └── pomiar_czasu
        ├── libpomiar_czasu.a
        ├── pomiar_czasu.c
        ├── pomiar_czasu.h
        ├── pomiar_czasu.o
        └── pomiar_czasu.tgz
```

---

Liczba iteracji - 10000000

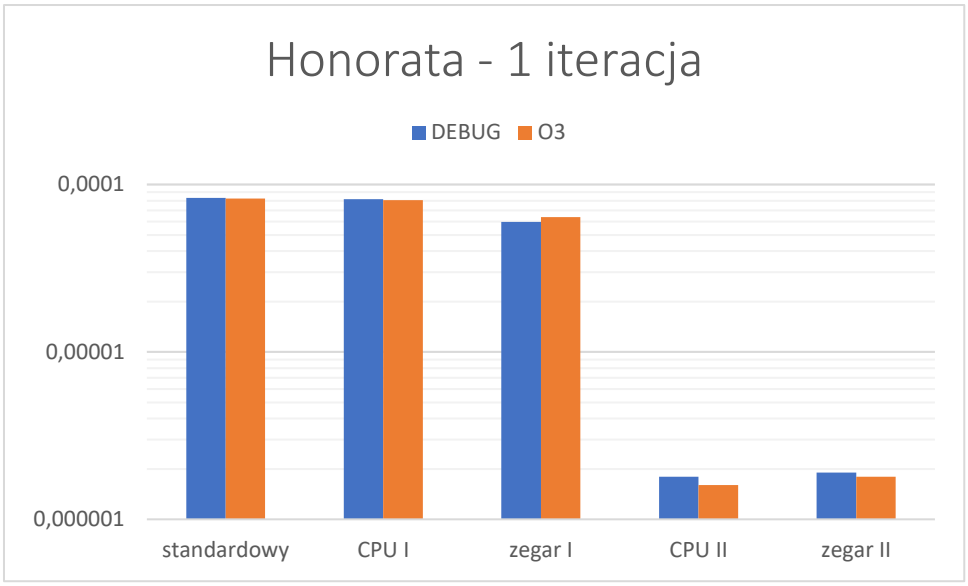
| -O3 | | Operacje we/wy | | Operacje arytmetyczne | |
|---|---|---|---|---|---|
| HONORATA | standardowy | CPU I | zegar I | CPU II | zegar II |
| 1 | 2,834371 | 2,145296 | 38,280211 | 0,049101 | 0,049099 |
| 2 | 2,823712 | 2,160439 | 41,026277 | 0,048796 | 0,0486 |
| 3 | 2,827539 | 2,151545 | 50,837176 | 0,048984 | 0,048983 |
| 4 | 2,82298 | 2,131112 | 46,174769 | 0,048692 | 0,048658 |
| 5 | 2,836379 | 2,089849 | 46,37539 | 0,048832 | 0,048333 |
| 6 | 2,824944 | 2,15694 | 46,654756 | 0,048838 | 0,048511 |
| 7 | 2,825597 | 2,015661 | 46,553575 | 0,04897 | 0,048729 |
| 8 | 2,84172 | 2,140881 | 45,870558 | 0,048712 | 0,048422 |
| 9 | 2,813919 | 2,13651 | 43,113074 | 0,049322 | 0,049321 |
| 10 | 2,937659 | 2,181171 | 41,651027 | 0,048604 | 0,048331 |
| średnia | 2,838882 | 2,1309404 | 44,6536813 | 0,0488851 | 0,0486987 |

| -g -DDEBUG | | Operacje we/wy | | Operacje arytmetyczne | |
|---|---|---|---|---|---|
| HONORATA | standardowy | CPU I | zegar I | CPU II | zegar II |
| 1 | 1,82543 | 1,317659 | 46,080735 | 0,05796 | 0,073397 |
| 2 | 2,970579 | 2,245678 | 44,124352 | 0,080545 | 0,080787 |
| 3 | 2,862436 | 2,1091 | 37,901801 | 0,079556 | 0,079751 |
| 4 | 3,156857 | 2,442886 | 40,413131 | 0,07999 | 0,080425 |
| 5 | 2,868208 | 2,180681 | 36,482889 | 0,080169 | 0,080379 |
| 6 | 2,952023 | 2,191823 | 39,243276 | 0,079827 | 0,079905 |
| 7 | 2,855454 | 2,18493 | 36,654453 | 0,080032 | 0,080235 |
| 8 | 2,856238 | 2,135123 | 41,439395 | 0,081073 | 0,081076 |
| 9 | 2,863002 | 2,156835 | 37,779845 | 0,082188 | 0,08219 |
| 10 | 2,858691 | 2,174217 | 38,200493 | 0,079714 | 0,080031 |
| średnia | 2,8068918 | 2,1138932 | 39,832037 | 0,0781054 | 0,0798176 |




Liczba iteracji – 1

| -g -DDEBUG | | Operacje we/wy | | Operacje arytmetyczne | |
|---|---|---|---|---|---|
| HONORATA | standardowy | CPU I | zegar I | CPU II | zegar II |
| 1 | 0,000072 | 0,000071 | 0,000057 | 0,000002 | 0,000002 |
| 2 | 0,000093 | 0,000091 | 0,000061 | 0,000002 | 0,000002 |
| 3 | 0,000084 | 0,000082 | 0,000052 | 0,000002 | 0,000002 |
| 4 | 0,000047 | 0,000047 | 0,000036 | 0,000001 | 0,000001 |
| 5 | 0,000082 | 0,00008 | 0,000066 | 0,000002 | 0,000002 |
| 6 | 0,000094 | 0,000092 | 0,000062 | 0,000001 | 0,000002 |
| 7 | 0,000082 | 0,000081 | 0,000067 | 0,000002 | 0,000002 |
| 8 | 0,000099 | 0,000098 | 0,000068 | 0,000002 | 0,000002 |
| 9 | 0,000077 | 0,000075 | 0,000058 | 0,000002 | 0,000002 |
| 10 | 0,000103 | 0,000102 | 0,000072 | 0,000002 | 0,000002 |
| średnia | 0,0000833 | 0,0000819 | 0,0000599 | 0,0000018 | 0,0000019 |




Wnioski:

Czas standardowy - odnosi się do rzeczywistego czasu, jaki upłynął od rozpoczęcia działania programu do momentu pomiaru czasu.
Czas CPU - odnosi się do czasu zużytego przez procesor na wykonywanie instrukcji programu, pomijając inne opóźnienia i czynniki związane z zegarem.
Czas zegarowy - odnosi się do czasu, który upłynął według zegara systemowego od momentu pomiaru czasu, jest to pomiar czasu na poziomie systemu operacyjnego i uwzględnia wszystkie opóźnienia, takie jak przełączanie kontekstu między procesami.

Dla operacji wejścia/wyjścia przy wykorzystaniu opcji do debugowania pomiar czasu wskazał czas wykonania ok. 25 razy większy dla czasu CPU oraz ok. 500 razy większy dla czasu zegara niż w przypadku wykonywania operacji arytmetycznych.

Oznacza to że operacje we/wy zajmują znacznie więcej czasu. Jest to spowodowane tym, że czas wykonania jest mierzony od momentu rozpoczęcia operacji odczytu/zapisu danych do urządzenia pamięci masowej, do momentu zakończenia operacji, gdy dane są dostępne w odpowiedniej pamięci lub gdy komunikacja z urządzeniem zewnętrznym zostaje ukończona. W efekcie podczas słabego połączenia z serwerem, czas może znacznie wzrosnąć.

Czas wykonania operacji arytmetycznych jest krótszy, gdyż mierzy on od rozpoczęcia obliczeń do momentu uzyskania wyniku tych obliczeń.

Średni czas wykonania pojedynczej operacji drukowania wynosi: 8,19E-05s (CPU) i 5,99E-05s (zegar)
Średni czas wykonania pojedynczej operacji arytmetycznej wynosi: 1,8E-06s (CPU) i 1,9E-06s (zegar)

Czas podczas kompilacji za pomocą wersji zoptymalizowanej do mierzenia czasu ulega zmianie.
Czas potrzebny na wykonanie operacji we/wy wzrasta średnio o 0,81% (CPU) i 12,1% (zegar).
Czas potrzebny na wykonanie operacji arytmetycznej maleje średnio o 37,41%(CPU) i 38,99%(zegar).

Optymalna wersja znacznie skraca czas potrzebny na wykonanie operacji arytmetycznych, nieznacznie wpływają na czas wykonania operacji we/wy.