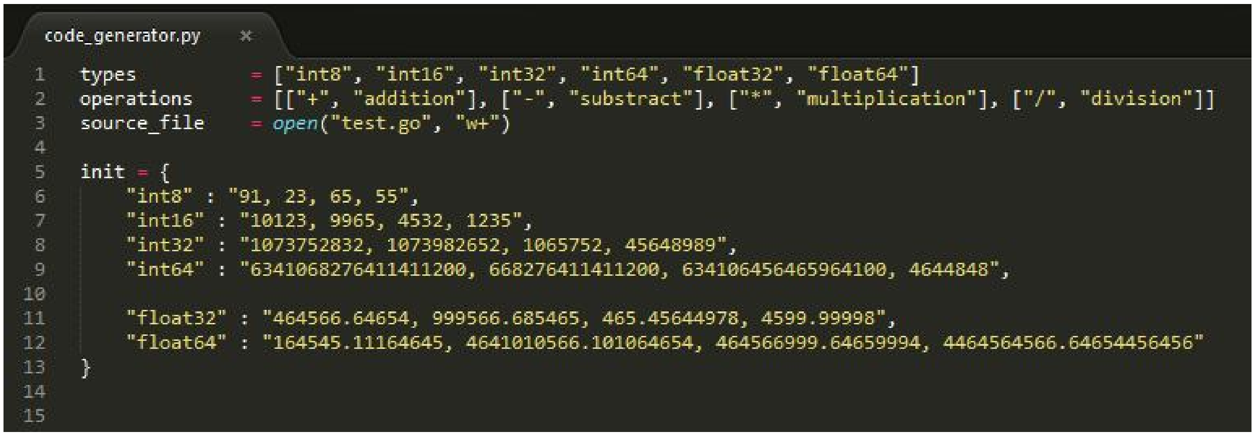
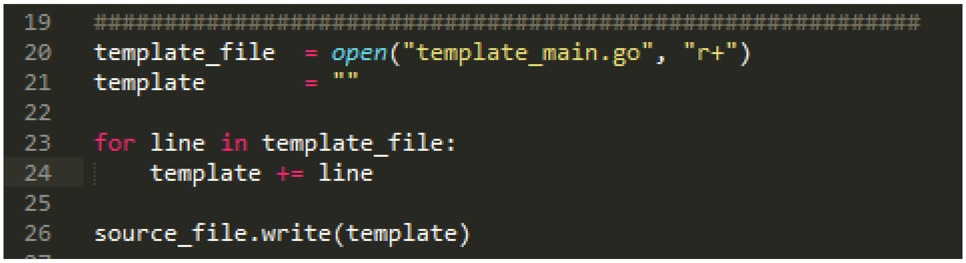
Звіт по лабораторній роботій №1
з архітектури обчислювальних систем
студента групи К22
Ламзіна Олега

1. В якості мови програмування для тестування швидкодії обч. системи було обрано мову програмування Go 1.5.
2. Крім того я написав скрипт на мові програмування Python, за допомогою якого генерується код на мові програмування Go, шляхом підстановки відповідних типів, значень та операцій в заготовку на Go.
3. Тести проводилися на двох обчислювальних системах:
   - Windows8 64bit, i3-3220 3.3GHz
   - Android 4.1.2, IdeaTab A1000L-F, Dual-core 1.2 GHz Cortex-A9

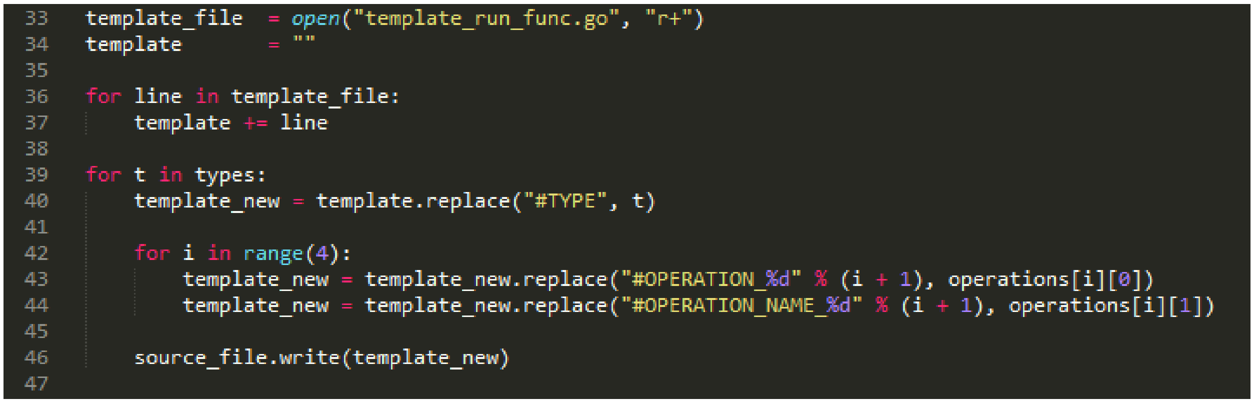
Лістинг скрипту:



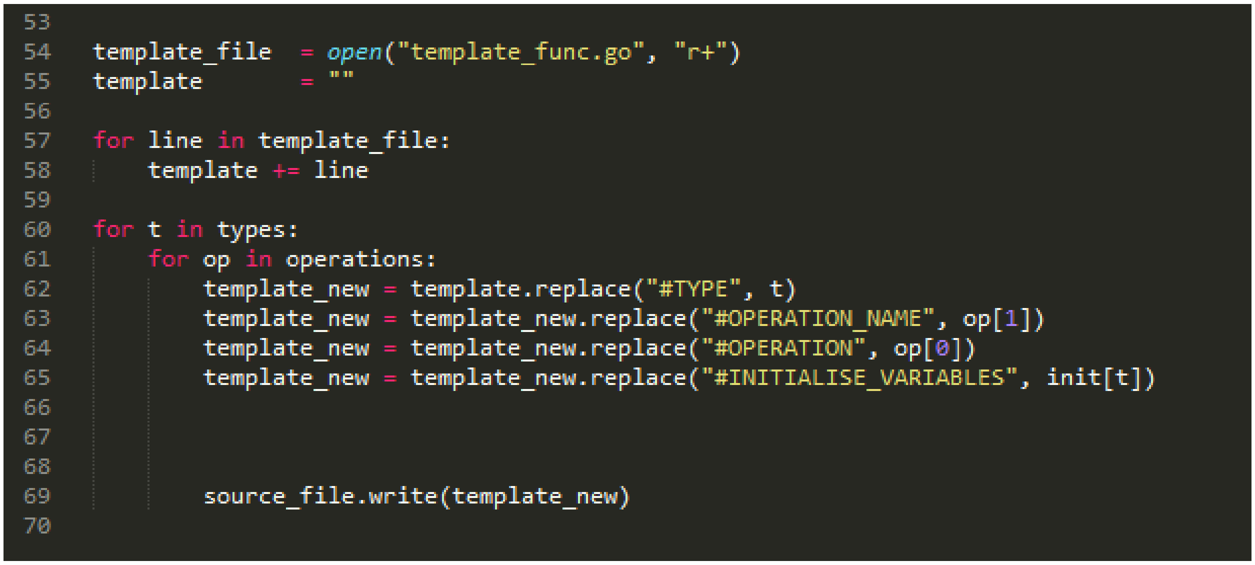
```python
types           = ["int8", "int16", "int32", "int64", "float32", "float64"]
operations      = [["+", "addition"], ["-", "substract"], ["*", "multiplication"], ["/", "division"]]
source_file     = open("test.go", "w+")

init = {
    "int8" : "91, 23, 65, 55",
    "int16" : "10123, 9965, 4532, 1235",
    "int32" : "1073752832, 1073982652, 1065752, 45648989",
    "int64" : "6341068276411411200, 668276411411200, 634106456465964100, 4644848",

    "float32" : "464566.64654, 999566.685465, 465.45644978, 4599.99998",
    "float64" : "164545.11164645, 4641010566.101064654, 464566999.64659994, 4464564566.64654456456"
}


```



```python
###########################################################
template_file  = open("template_main.go", "r+")
template       = ""

for line in template_file:
    template += line

source_file.write(template)
```



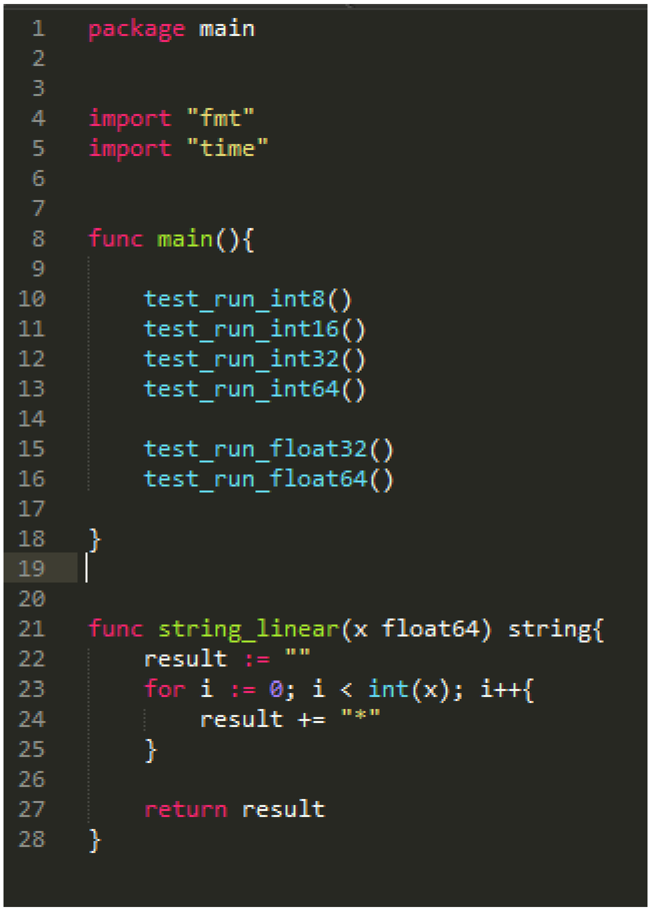
```python
template_file  = open("template_run_func.go", "r+")
template       = ""

for line in template_file:
    template += line

for t in types:
    template_new = template.replace("#TYPE", t)

    for i in range(4):
        template_new = template_new.replace("#OPERATION_%d" % (i + 1), operations[i][0])
        template_new = template_new.replace("#OPERATION_NAME_%d" % (i + 1), operations[i][1])

    source_file.write(template_new)

```



```python

template_file  = open("template_func.go", "r+")
template       = ""

for line in template_file:
    template += line

for t in types:
    for op in operations:
        template_new = template.replace("#TYPE", t)
        template_new = template_new.replace("#OPERATION_NAME", op[1])
        template_new = template_new.replace("#OPERATION", op[0])
        template_new = template_new.replace("#INITIALISE_VARIABLES", init[t])



        source_file.write(template_new)

```

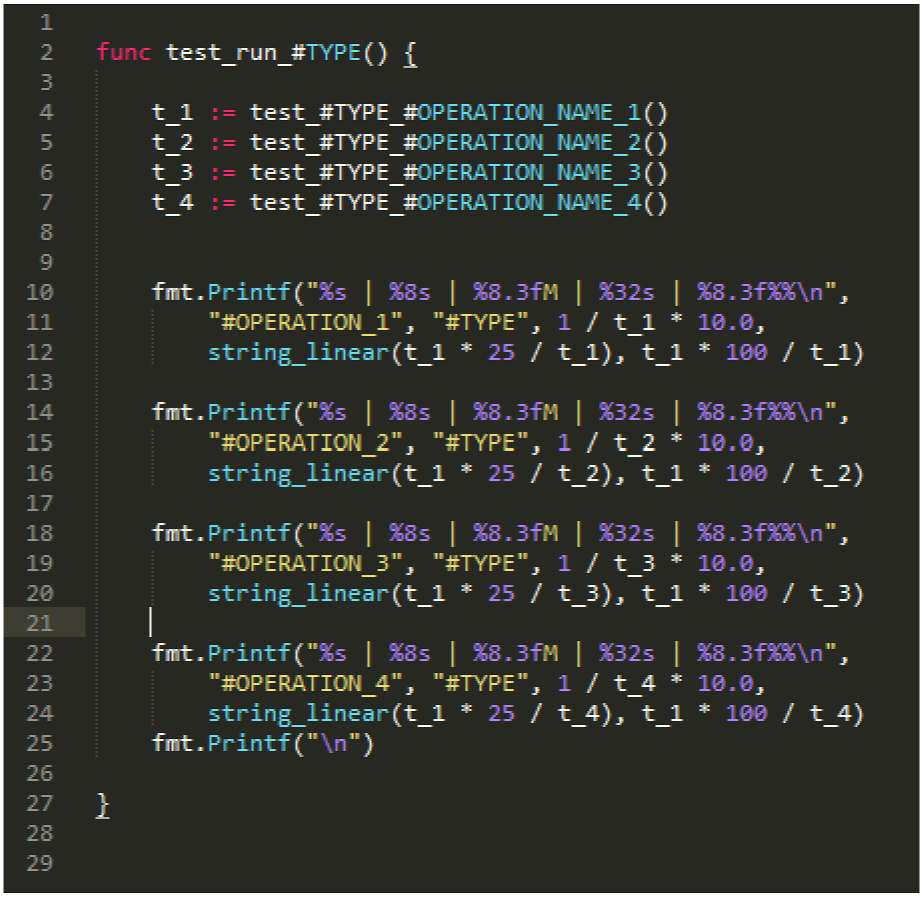
"code_generator.py" створює "test.go" з файлів "template_func.go", "template_main.go" & "template_run_func.go"

"template_main.go" лістинг:



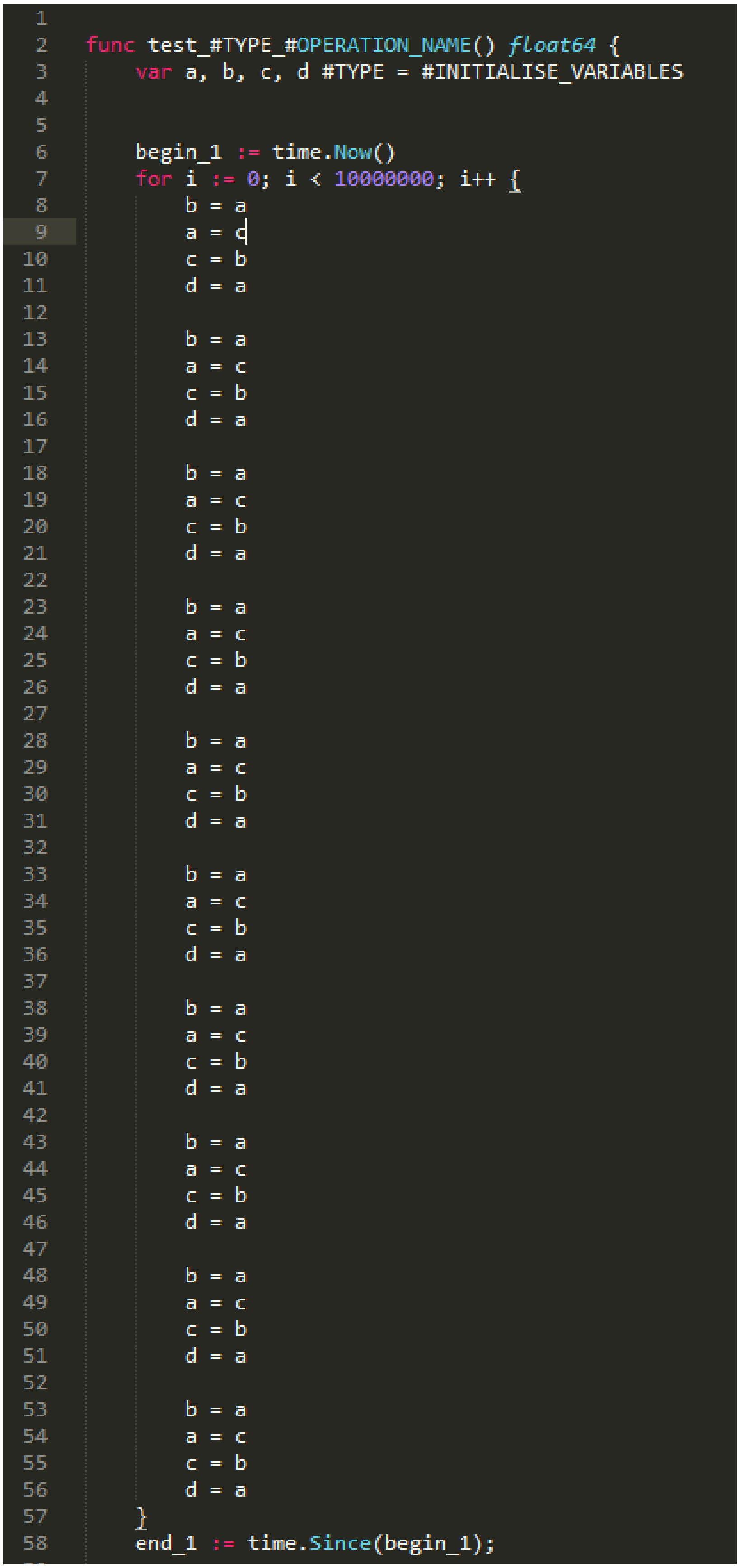
```go
package main


import "fmt"
import "time"


func main(){

    test_run_int8()
    test_run_int16()
    test_run_int32()
    test_run_int64()

    test_run_float32()
    test_run_float64()

}


func string_linear(x float64) string{
    result := ""
    for i := 0; i < int(x); i++{
        result += "*"
    }

    return result
}
```

"template_run_func.go" лістинг:

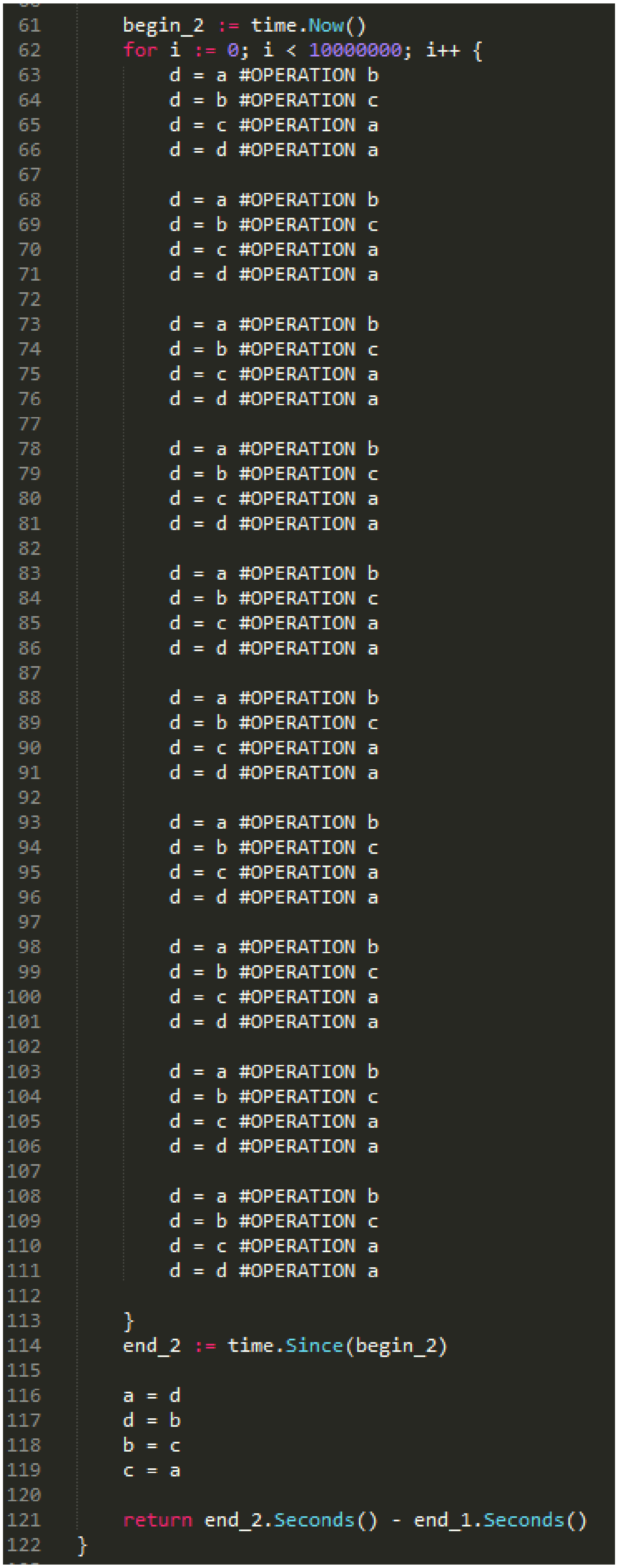


```
func test_run_#TYPE() {

    t_1 := test_#TYPE_#OPERATION_NAME_1()
    t_2 := test_#TYPE_#OPERATION_NAME_2()
    t_3 := test_#TYPE_#OPERATION_NAME_3()
    t_4 := test_#TYPE_#OPERATION_NAME_4()


    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
        "#OPERATION_1", "#TYPE", 1 / t_1 * 10.0,
        string_linear(t_1 * 25 / t_1), t_1 * 100 / t_1)

    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
        "#OPERATION_2", "#TYPE", 1 / t_2 * 10.0,
        string_linear(t_1 * 25 / t_2), t_1 * 100 / t_2)

    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
        "#OPERATION_3", "#TYPE", 1 / t_3 * 10.0,
        string_linear(t_1 * 25 / t_3), t_1 * 100 / t_3)

    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
        "#OPERATION_4", "#TYPE", 1 / t_4 * 10.0,
        string_linear(t_1 * 25 / t_4), t_1 * 100 / t_4)
    fmt.Printf("\n")

}
```

“template_func.go” лістинг:



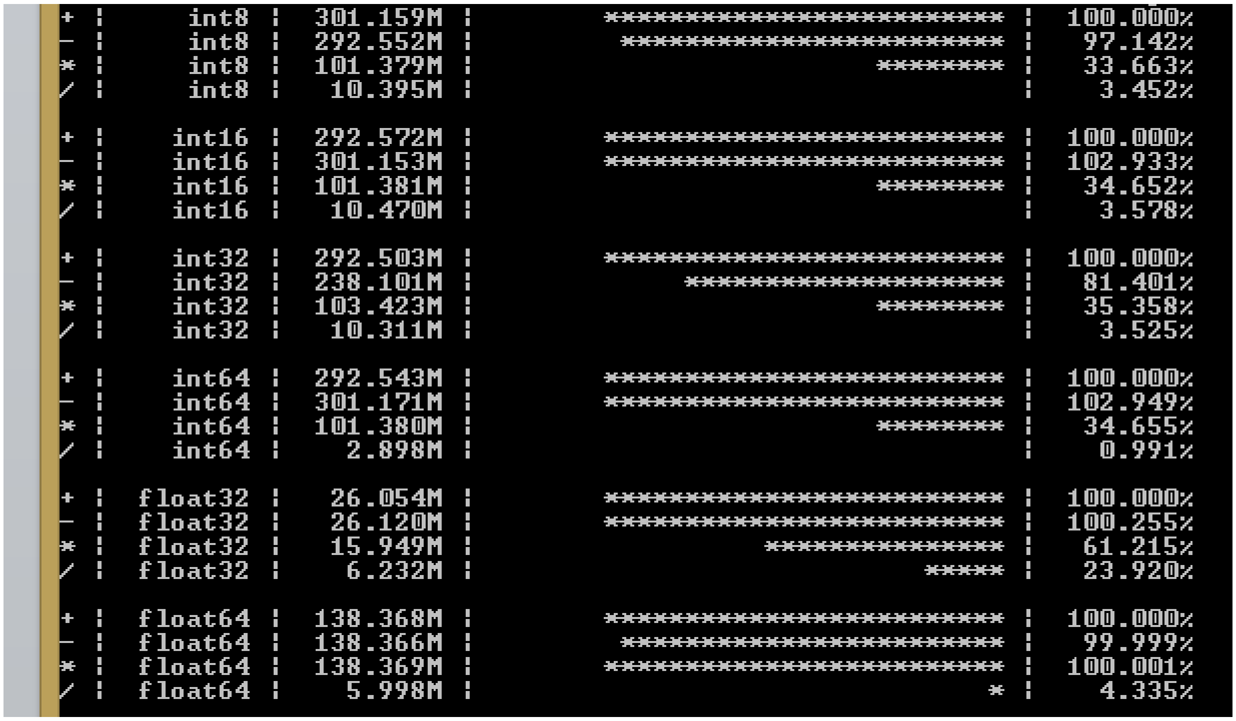
```go

func test_#TYPE_#OPERATION_NAME() float64 {
    var a, b, c, d #TYPE = #INITIALISE_VARIABLES


    begin_1 := time.Now()
    for i := 0; i < 10000000; i++ {
        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a

        b = a
        a = c
        c = b
        d = a
    }
    end_1 := time.Since(begin_1);
```



```
    begin_2 := time.Now()
    for i := 0; i < 10000000; i++ {
        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

        d = a #OPERATION b
        d = b #OPERATION c
        d = c #OPERATION a
        d = d #OPERATION a

    }
    end_2 := time.Since(begin_2)

    a = d
    d = b
    b = c
    c = a

    return end_2.Seconds() - end_1.Seconds()
}
```

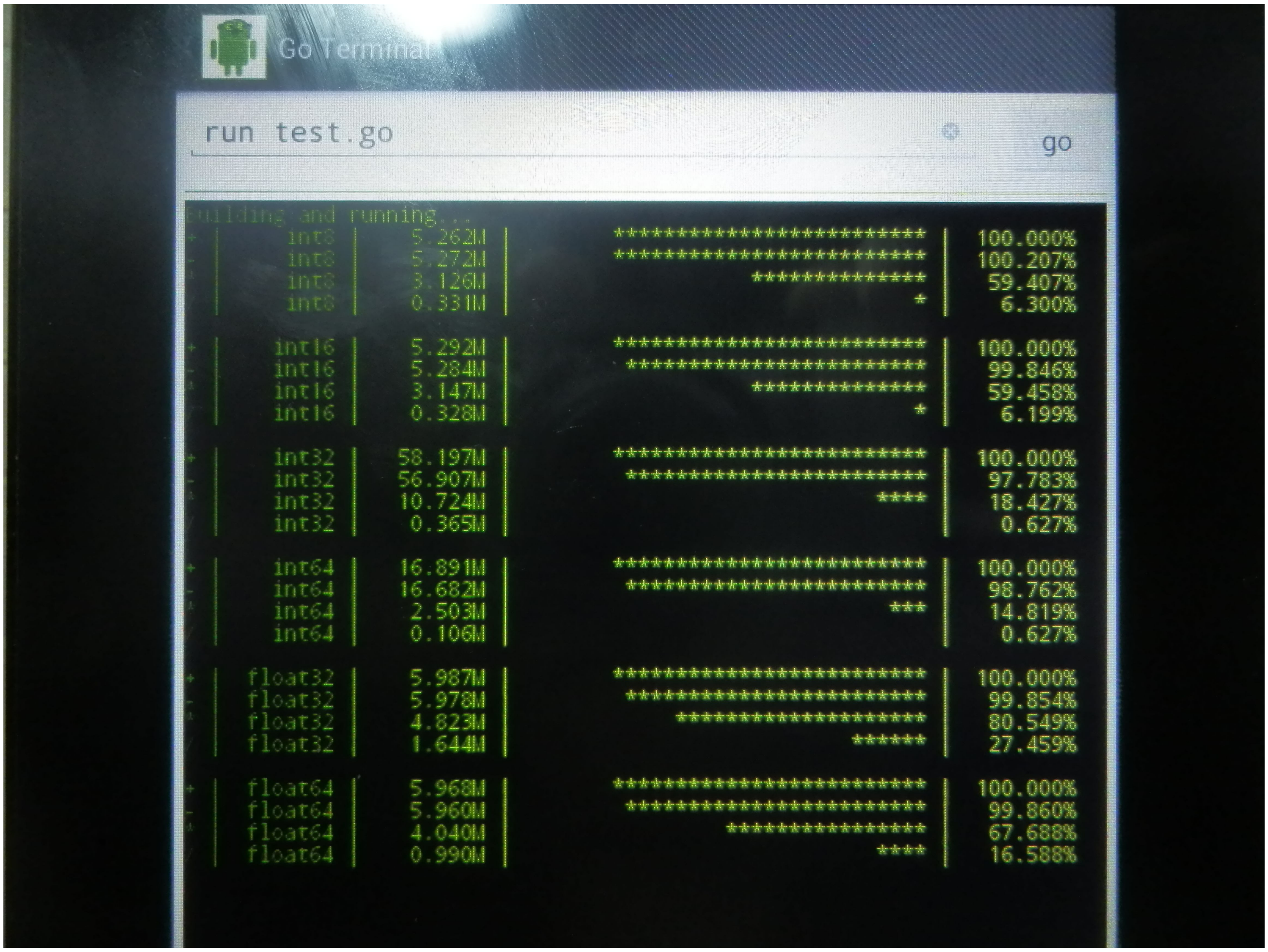
Результати:

Windows8, i3:



```
+ ¦     int8 ¦  301.159M ¦      ************************* ¦ 100.000%
- ¦     int8 ¦  292.552M ¦       ************************ ¦  97.142%
* ¦     int8 ¦  101.379M ¦                       ******** ¦  33.663%
/ ¦     int8 ¦   10.395M ¦                                ¦   3.452%

+ ¦    int16 ¦  292.572M ¦      ************************* ¦ 100.000%
- ¦    int16 ¦  301.153M ¦      ************************* ¦ 102.933%
* ¦    int16 ¦  101.381M ¦                       ******** ¦  34.652%
/ ¦    int16 ¦   10.470M ¦                                ¦   3.578%

+ ¦    int32 ¦  292.503M ¦      ************************* ¦ 100.000%
- ¦    int32 ¦  238.101M ¦           ******************** ¦  81.401%
* ¦    int32 ¦  103.423M ¦                       ******** ¦  35.358%
/ ¦    int32 ¦   10.311M ¦                                ¦   3.525%

+ ¦    int64 ¦  292.543M ¦      ************************* ¦ 100.000%
- ¦    int64 ¦  301.171M ¦      ************************* ¦ 102.949%
* ¦    int64 ¦  101.380M ¦                       ******** ¦  34.655%
/ ¦    int64 ¦    2.898M ¦                                ¦   0.991%

+ ¦  float32 ¦   26.054M ¦      ************************* ¦ 100.000%
- ¦  float32 ¦   26.120M ¦      ************************* ¦ 100.255%
* ¦  float32 ¦   15.949M ¦                 *************** ¦  61.215%
/ ¦  float32 ¦    6.232M ¦                          ***** ¦  23.920%

+ ¦  float64 ¦  138.368M ¦      ************************* ¦ 100.000%
- ¦  float64 ¦  138.366M ¦       ************************ ¦  99.999%
* ¦  float64 ¦  138.369M ¦      ************************* ¦ 100.001%
/ ¦  float64 ¦    5.998M ¦                              * ¦   4.335%
```

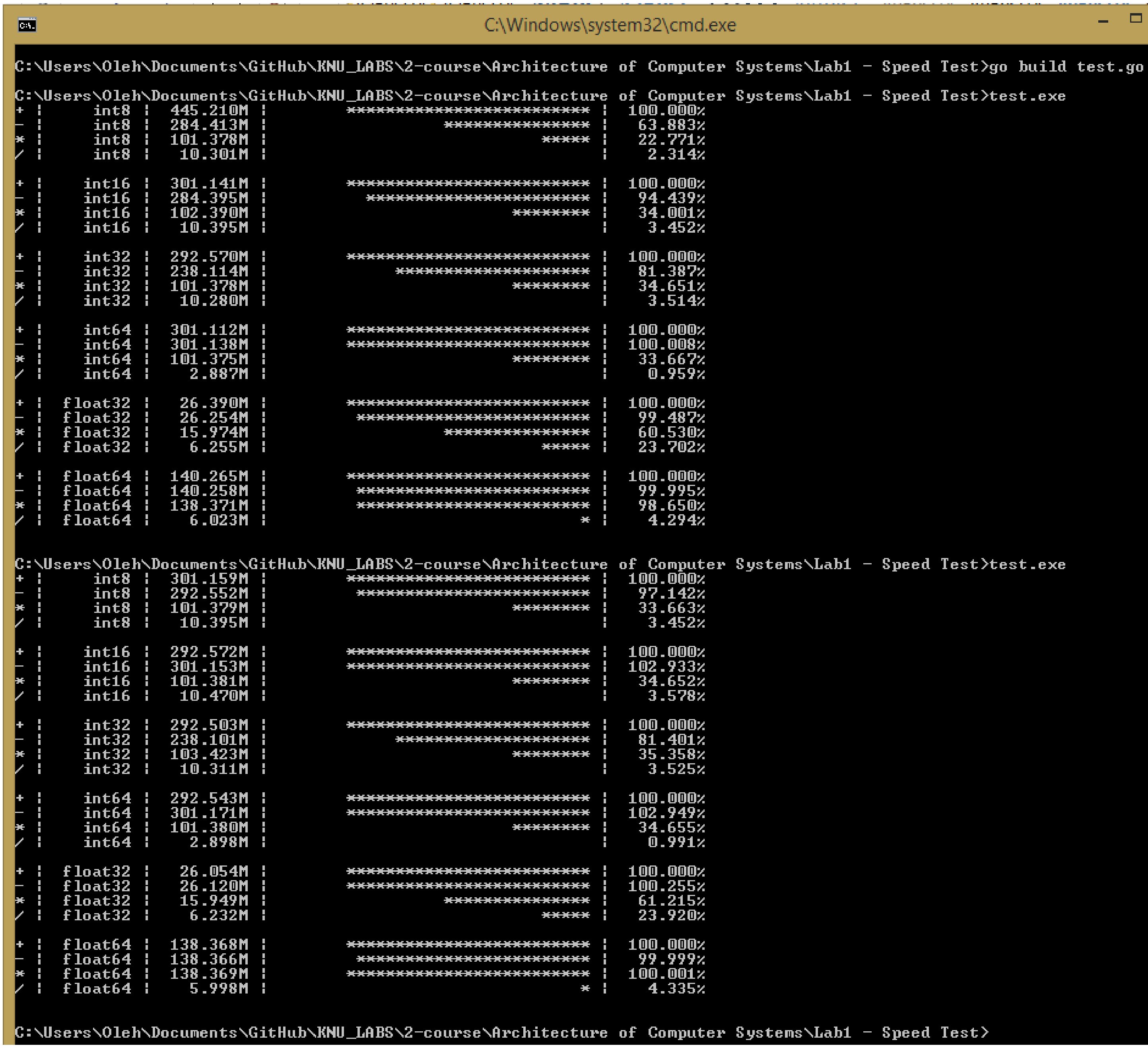
Android 4.1.2, Dual-core 1.2 GHz Cortex-A9



```
run test.go
```

```
Building and running...
+ |     int8 |    5.262M |      ************************* | 100.000%
- |     int8 |    5.272M |      ************************* | 100.207%
* |     int8 |    3.126M |                 ************** |  59.407%
/ |     int8 |    0.331M |                              * |   6.300%

+ |    int16 |    5.292M |      ************************* | 100.000%
- |    int16 |    5.284M |       ************************ |  99.846%
* |    int16 |    3.147M |                 ************** |  59.458%
/ |    int16 |    0.328M |                              * |   6.199%

+ |    int32 |   58.197M |      ************************* | 100.000%
- |    int32 |   56.907M |       ************************ |  97.783%
* |    int32 |   10.724M |                           **** |  18.427%
/ |    int32 |    0.365M |                                |   0.627%

+ |    int64 |   16.891M |      ************************* | 100.000%
- |    int64 |   16.682M |       ************************ |  98.762%
* |    int64 |    2.503M |                            *** |  14.819%
/ |    int64 |    0.106M |                                |   0.627%

+ |  float32 |    5.987M |      ************************* | 100.000%
- |  float32 |    5.978M |       ************************ |  99.854%
* |  float32 |    4.823M |           ******************** |  80.549%
/ |  float32 |    1.644M |                         ****** |  27.459%

+ |  float64 |    5.968M |      ************************* | 100.000%
- |  float64 |    5.960M |       ************************ |  99.860%
* |  float64 |    4.040M |               **************** |  67.688%
/ |  float64 |    0.990M |                           **** |  16.588%
```

Слід додати також те що тест веде себе стабільно на різних запусках:



```
C:\Users\Oleh\Documents\GitHub\KNU_LABS\2-course\Architecture of Computer Systems\Lab1 - Speed Test>go build test.go

C:\Users\Oleh\Documents\GitHub\KNU_LABS\2-course\Architecture of Computer Systems\Lab1 - Speed Test>test.exe
+ ¦     int8 ¦  445.210M ¦        ************************* ¦  100.000%
- ¦     int8 ¦  284.413M ¦                  *************** ¦   63.883%
* ¦     int8 ¦  101.378M ¦                            ***** ¦   22.771%
/ ¦     int8 ¦   10.301M ¦                                  ¦    2.314%

+ ¦    int16 ¦  301.141M ¦        ************************* ¦  100.000%
- ¦    int16 ¦  284.395M ¦          *********************** ¦   94.439%
* ¦    int16 ¦  102.390M ¦                         ******** ¦   34.001%
/ ¦    int16 ¦   10.395M ¦                                  ¦    3.452%

+ ¦    int32 ¦  292.570M ¦        ************************* ¦  100.000%
- ¦    int32 ¦  238.114M ¦             ******************** ¦   81.387%
* ¦    int32 ¦  101.378M ¦                         ******** ¦   34.651%
/ ¦    int32 ¦   10.280M ¦                                  ¦    3.514%

+ ¦    int64 ¦  301.112M ¦        ************************* ¦  100.000%
- ¦    int64 ¦  301.138M ¦        ************************* ¦  100.008%
* ¦    int64 ¦  101.375M ¦                         ******** ¦   33.667%
/ ¦    int64 ¦    2.887M ¦                                  ¦    0.959%

+ ¦  float32 ¦   26.390M ¦        ************************* ¦  100.000%
- ¦  float32 ¦   26.254M ¦         ************************ ¦   99.487%
* ¦  float32 ¦   15.974M ¦                  *************** ¦   60.530%
/ ¦  float32 ¦    6.255M ¦                            ***** ¦   23.702%

+ ¦  float64 ¦  140.265M ¦        ************************* ¦  100.000%
- ¦  float64 ¦  140.258M ¦         ************************ ¦   99.995%
* ¦  float64 ¦  138.371M ¦         ************************ ¦   98.650%
/ ¦  float64 ¦    6.023M ¦                                * ¦    4.294%


C:\Users\Oleh\Documents\GitHub\KNU_LABS\2-course\Architecture of Computer Systems\Lab1 - Speed Test>test.exe
+ ¦     int8 ¦  301.159M ¦        ************************* ¦  100.000%
- ¦     int8 ¦  292.552M ¦         ************************ ¦   97.142%
* ¦     int8 ¦  101.379M ¦                         ******** ¦   33.663%
/ ¦     int8 ¦   10.395M ¦                                  ¦    3.452%

+ ¦    int16 ¦  292.572M ¦        ************************* ¦  100.000%
- ¦    int16 ¦  301.153M ¦        ************************* ¦  102.933%
* ¦    int16 ¦  101.381M ¦                         ******** ¦   34.652%
/ ¦    int16 ¦   10.470M ¦                                  ¦    3.578%

+ ¦    int32 ¦  292.503M ¦        ************************* ¦  100.000%
- ¦    int32 ¦  238.101M ¦             ******************** ¦   81.401%
* ¦    int32 ¦  103.423M ¦                         ******** ¦   35.358%
/ ¦    int32 ¦   10.311M ¦                                  ¦    3.525%

+ ¦    int64 ¦  292.543M ¦        ************************* ¦  100.000%
- ¦    int64 ¦  301.171M ¦        ************************* ¦  102.949%
* ¦    int64 ¦  101.380M ¦                         ******** ¦   34.655%
/ ¦    int64 ¦    2.898M ¦                                  ¦    0.991%

+ ¦  float32 ¦   26.054M ¦        ************************* ¦  100.000%
- ¦  float32 ¦   26.120M ¦        ************************* ¦  100.255%
* ¦  float32 ¦   15.949M ¦                  *************** ¦   61.215%
/ ¦  float32 ¦    6.232M ¦                            ***** ¦   23.920%

+ ¦  float64 ¦  138.368M ¦        ************************* ¦  100.000%
- ¦  float64 ¦  138.366M ¦         ************************ ¦   99.999%
* ¦  float64 ¦  138.369M ¦        ************************* ¦  100.001%
/ ¦  float64 ¦    5.998M ¦                                * ¦    4.335%


C:\Users\Oleh\Documents\GitHub\KNU_LABS\2-course\Architecture of Computer Systems\Lab1 - Speed Test>
```