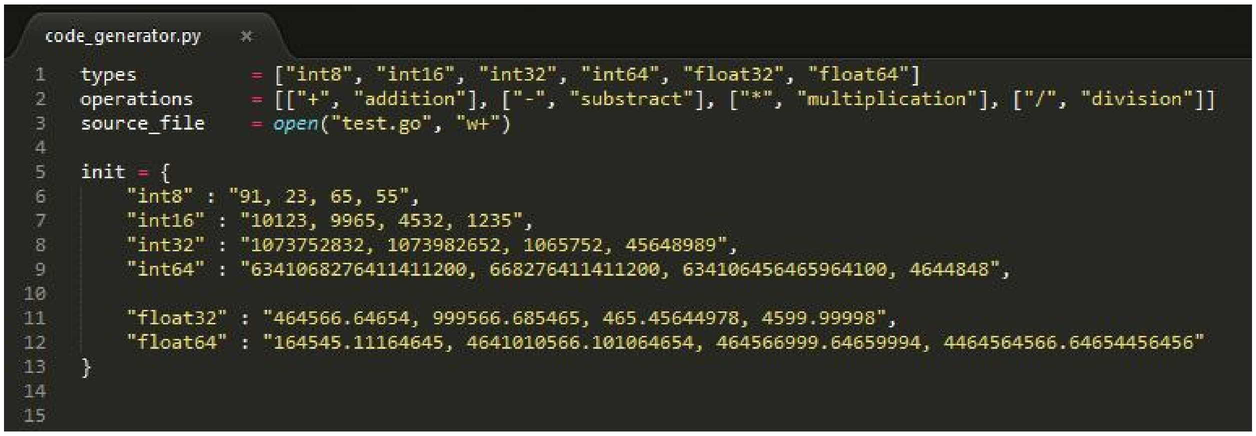
Звіт по лабораторній роботій №1
з архітектури обчислювальних систем

студента групи К22

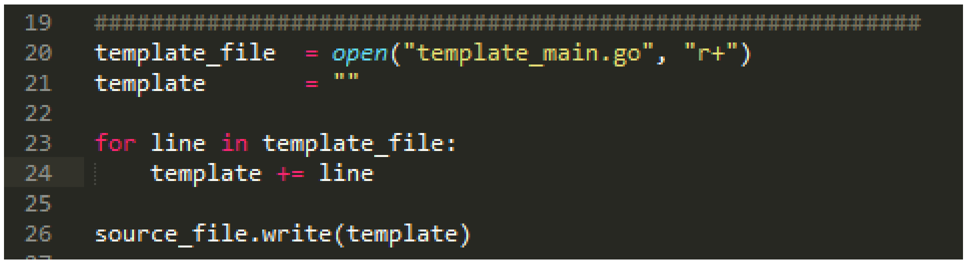
Ламзіна Олега

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

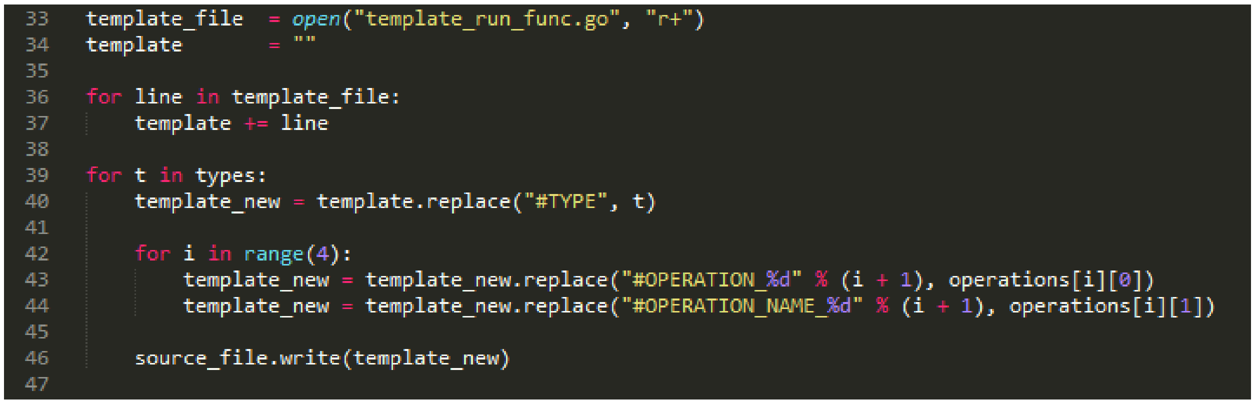
Лістинг скрипту:



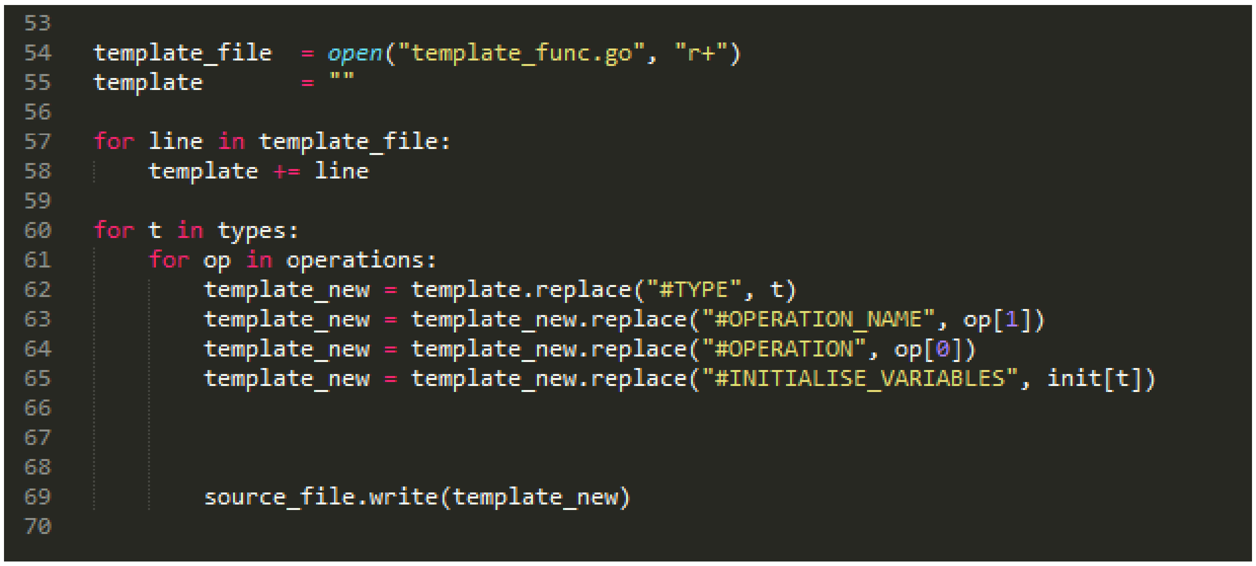
```python
types           = ["int8", "int16", "int32", "int64", "float32", "float64"]
operations      = [["+", "addition"], ["-", "substract"], ["*", "multiplication"], ["/", "division"]]
source_file     = open("test.go", "w+")

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)
```



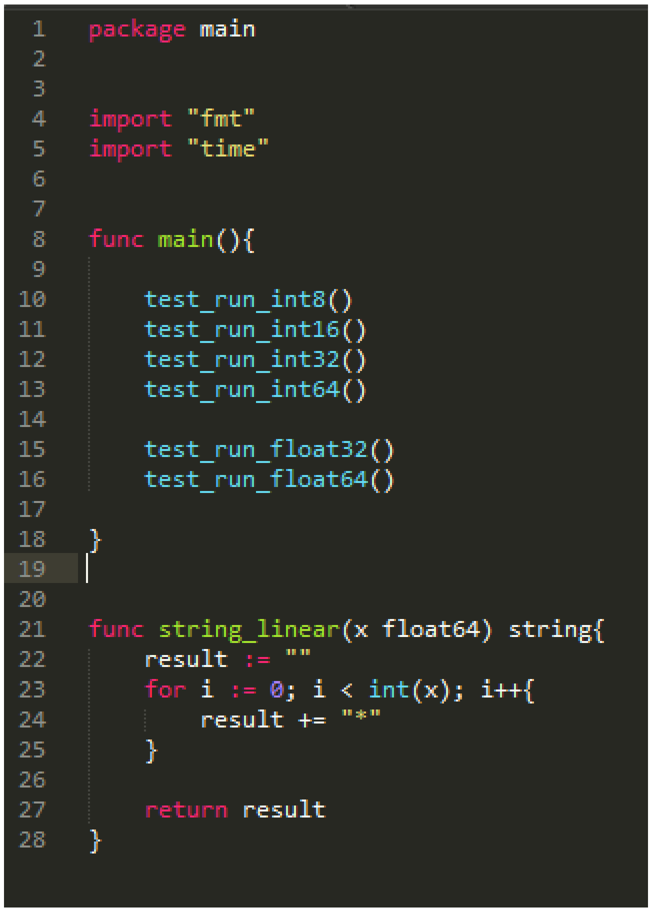
```python
template_file   = open("template_run_func.go", "r+")
template        = ""

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)

```

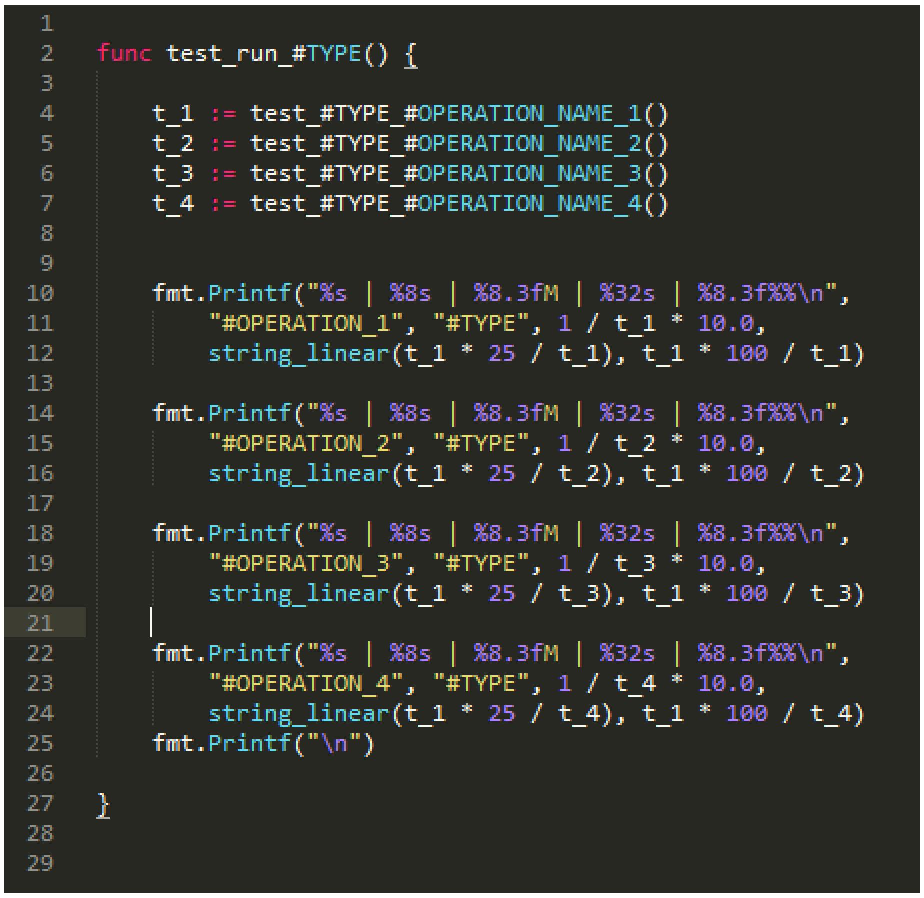
“code_generator.py” створює “test.go” з файлів “template_func.go”, “template_main.go” & “template_run_func.go”

“template_main.go” лістинг:



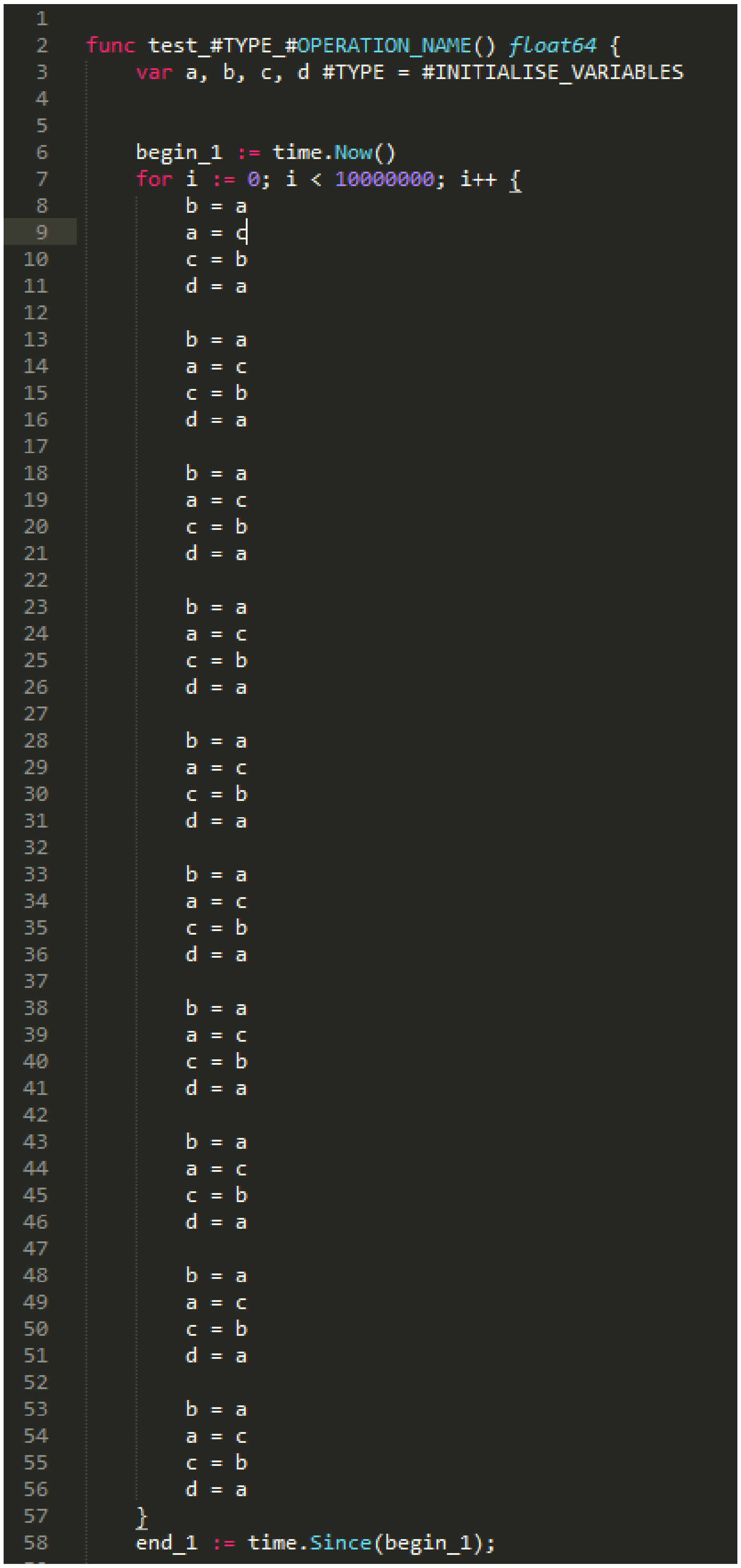
```go
package main


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:

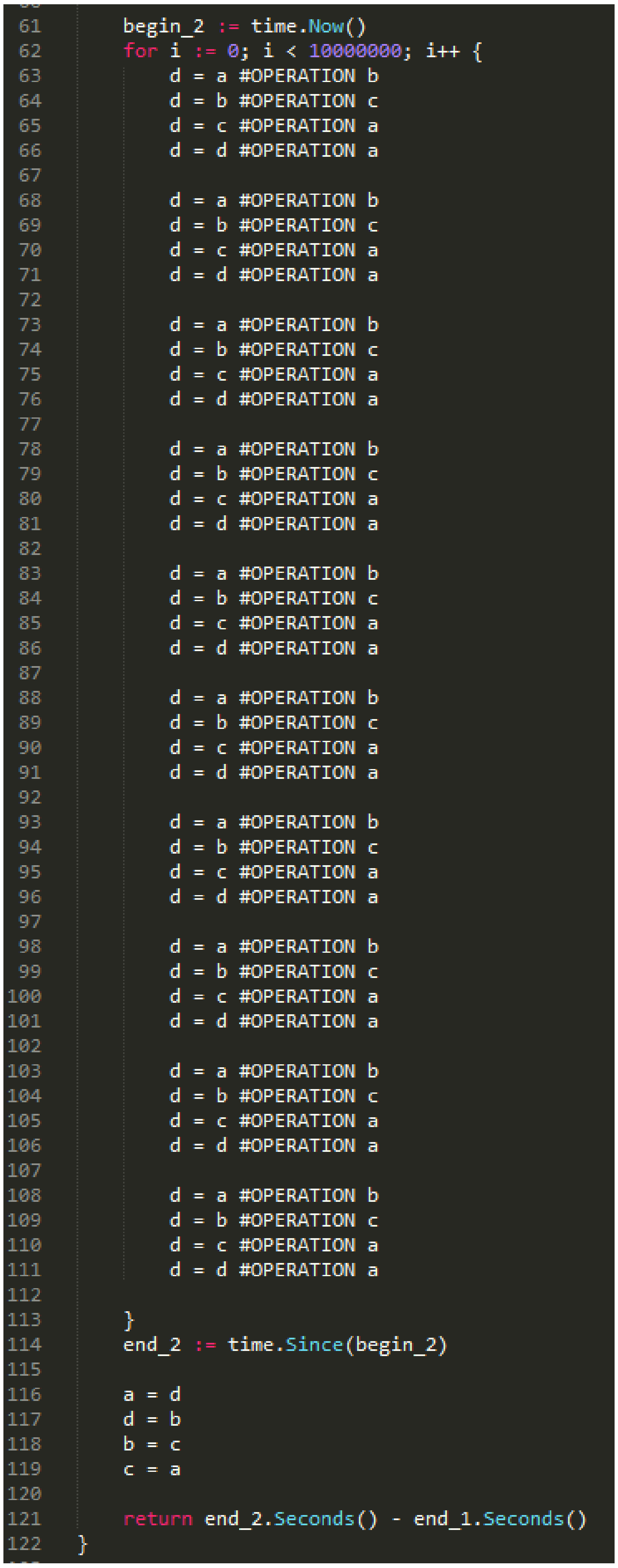


```go

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:

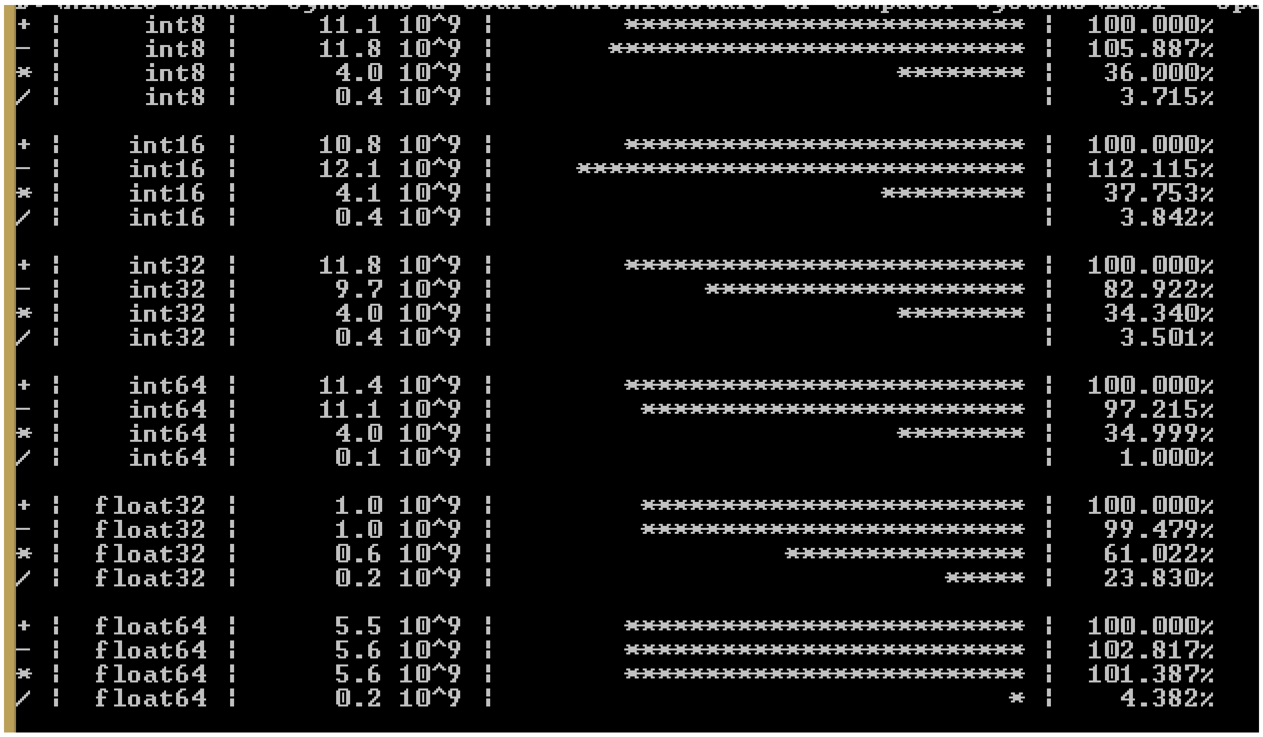


```

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()
}
```

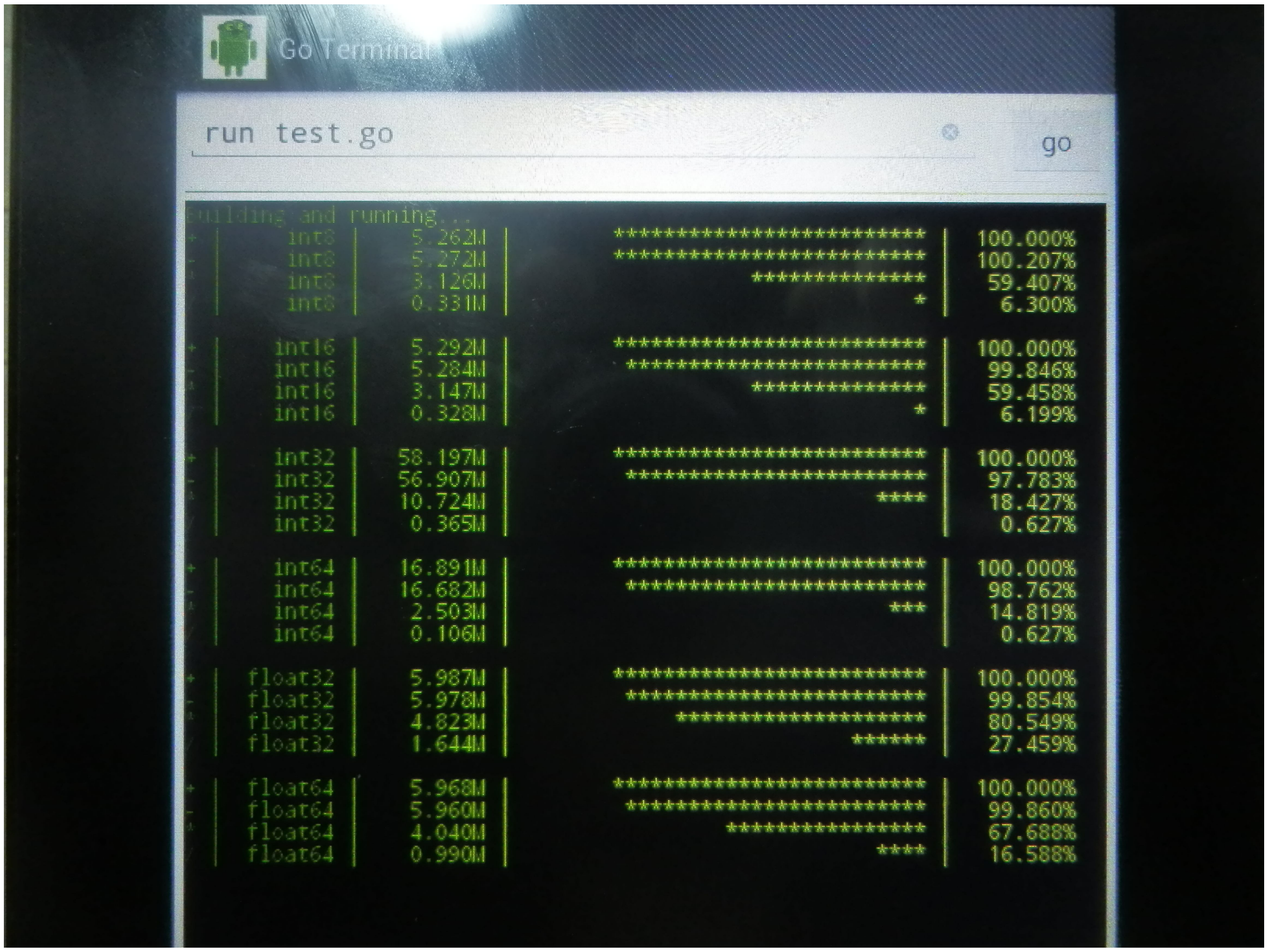
Результати:

Windows8, i3:



```
+ ¦    int8 ¦    11.1 10^9 ¦         ************************* ¦  100.000%
- ¦    int8 ¦    11.8 10^9 ¦        ************************** ¦  105.887%
* ¦    int8 ¦     4.0 10^9 ¦                         ********* ¦   36.000%
/ ¦    int8 ¦     0.4 10^9 ¦                                   ¦    3.715%

+ ¦   int16 ¦    10.8 10^9 ¦         ************************* ¦  100.000%
- ¦   int16 ¦    12.1 10^9 ¦      **************************** ¦  112.115%
* ¦   int16 ¦     4.1 10^9 ¦                         ********* ¦   37.753%
/ ¦   int16 ¦     0.4 10^9 ¦                                   ¦    3.842%

+ ¦   int32 ¦    11.8 10^9 ¦         ************************* ¦  100.000%
- ¦   int32 ¦     9.7 10^9 ¦              ******************** ¦   82.922%
* ¦   int32 ¦     4.0 10^9 ¦                          ******** ¦   34.340%
/ ¦   int32 ¦     0.4 10^9 ¦                                   ¦    3.501%

+ ¦   int64 ¦    11.4 10^9 ¦         ************************* ¦  100.000%
- ¦   int64 ¦    11.1 10^9 ¦          ************************ ¦   97.215%
* ¦   int64 ¦     4.0 10^9 ¦                          ******** ¦   34.999%
/ ¦   int64 ¦     0.1 10^9 ¦                                   ¦    1.000%

+ ¦ float32 ¦     1.0 10^9 ¦          ************************ ¦  100.000%
- ¦ float32 ¦     1.0 10^9 ¦          ************************ ¦   99.479%
* ¦ float32 ¦     0.6 10^9 ¦                   *************** ¦   61.022%
/ ¦ float32 ¦     0.2 10^9 ¦                             ***** ¦   23.830%

+ ¦ float64 ¦     5.5 10^9 ¦         ************************* ¦  100.000%
- ¦ float64 ¦     5.6 10^9 ¦         ************************* ¦  102.817%
* ¦ float64 ¦     5.6 10^9 ¦         ************************* ¦  101.387%
/ ¦ float64 ¦     0.2 10^9 ¦                                 * ¦    4.382%
```

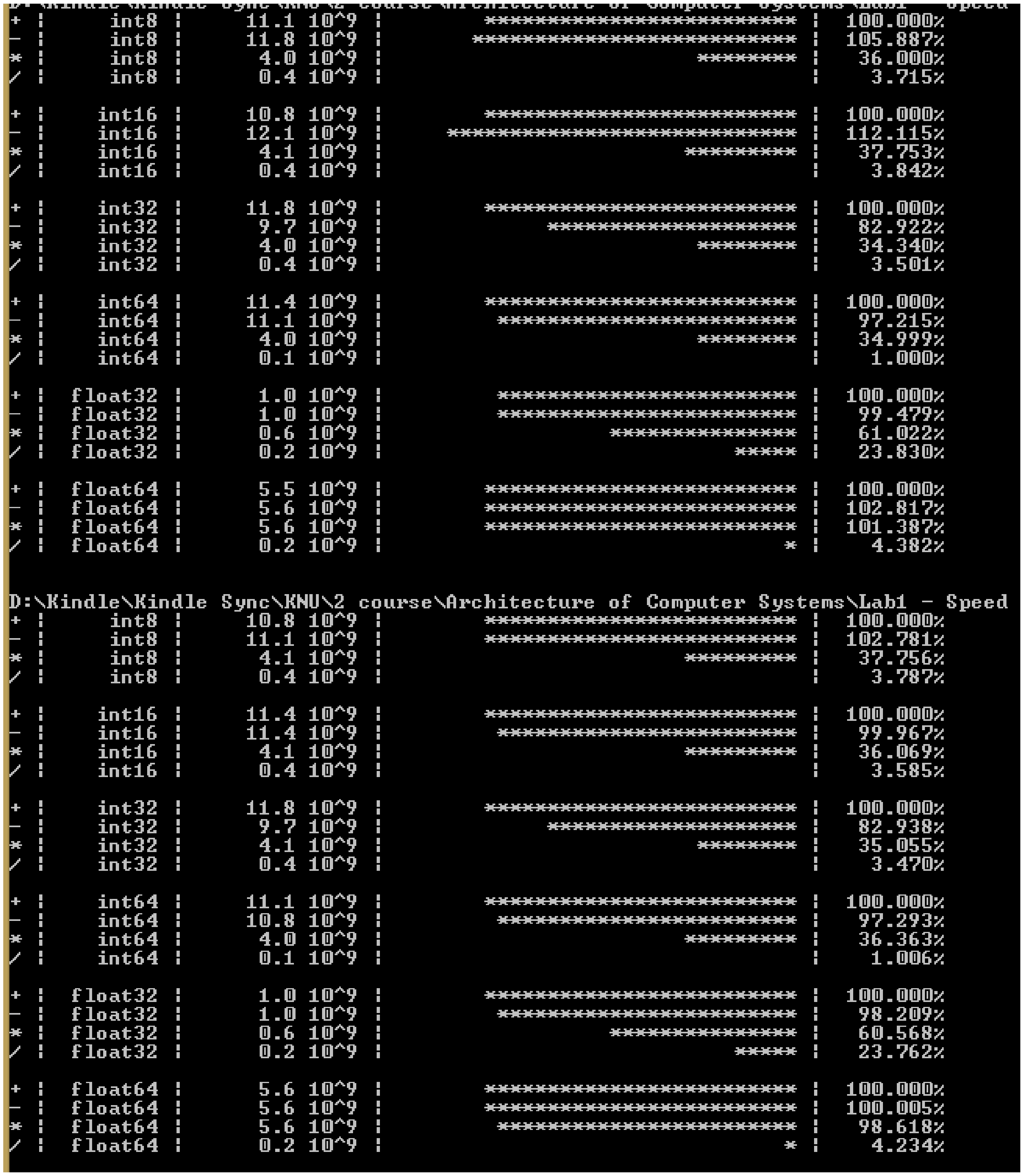
Android 4.1.2, Dual-core 1.2 GHz Cortex-A9



```
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:



```
D:\Kindle\Kindle Sync\KNU\2 course\Architecture of Computer Systems\Lab1 - Speed
+ ¦     int8 ¦     11.1 10^9 ¦         ************************* ¦  100.000%
- ¦     int8 ¦     11.8 10^9 ¦        ************************** ¦  105.887%
* ¦     int8 ¦      4.0 10^9 ¦                          ******** ¦   36.000%
/ ¦     int8 ¦      0.4 10^9 ¦                                  ¦    3.715%

+ ¦    int16 ¦     10.8 10^9 ¦         ************************* ¦  100.000%
- ¦    int16 ¦     12.1 10^9 ¦      **************************** ¦  112.115%
* ¦    int16 ¦      4.1 10^9 ¦                         ********* ¦   37.753%
/ ¦    int16 ¦      0.4 10^9 ¦                                  ¦    3.842%

+ ¦    int32 ¦     11.8 10^9 ¦         ************************* ¦  100.000%
- ¦    int32 ¦      9.7 10^9 ¦              ******************** ¦   82.922%
* ¦    int32 ¦      4.0 10^9 ¦                          ******** ¦   34.340%
/ ¦    int32 ¦      0.4 10^9 ¦                                  ¦    3.501%

+ ¦    int64 ¦     11.4 10^9 ¦         ************************* ¦  100.000%
- ¦    int64 ¦     11.1 10^9 ¦          ************************ ¦   97.215%
* ¦    int64 ¦      4.0 10^9 ¦                          ******** ¦   34.999%
/ ¦    int64 ¦      0.1 10^9 ¦                                  ¦    1.000%

+ ¦  float32 ¦      1.0 10^9 ¦          ************************ ¦  100.000%
- ¦  float32 ¦      1.0 10^9 ¦          ************************ ¦   99.479%
* ¦  float32 ¦      0.6 10^9 ¦                   *************** ¦   61.022%
/ ¦  float32 ¦      0.2 10^9 ¦                             ***** ¦   23.830%

+ ¦  float64 ¦      5.5 10^9 ¦         ************************* ¦  100.000%
- ¦  float64 ¦      5.6 10^9 ¦         ************************* ¦  102.817%
* ¦  float64 ¦      5.6 10^9 ¦         ************************* ¦  101.387%
/ ¦  float64 ¦      0.2 10^9 ¦                                 * ¦    4.382%


D:\Kindle\Kindle Sync\KNU\2 course\Architecture of Computer Systems\Lab1 - Speed
+ ¦     int8 ¦     10.8 10^9 ¦         ************************* ¦  100.000%
- ¦     int8 ¦     11.1 10^9 ¦         ************************* ¦  102.781%
* ¦     int8 ¦      4.1 10^9 ¦                         ********* ¦   37.756%
/ ¦     int8 ¦      0.4 10^9 ¦                                  ¦    3.787%

+ ¦    int16 ¦     11.4 10^9 ¦         ************************* ¦  100.000%
- ¦    int16 ¦     11.4 10^9 ¦          ************************ ¦   99.967%
* ¦    int16 ¦      4.1 10^9 ¦                         ********* ¦   36.069%
/ ¦    int16 ¦      0.4 10^9 ¦                                  ¦    3.585%

+ ¦    int32 ¦     11.8 10^9 ¦         ************************* ¦  100.000%
- ¦    int32 ¦      9.7 10^9 ¦              ******************** ¦   82.938%
* ¦    int32 ¦      4.1 10^9 ¦                          ******** ¦   35.055%
/ ¦    int32 ¦      0.4 10^9 ¦                                  ¦    3.470%

+ ¦    int64 ¦     11.1 10^9 ¦         ************************* ¦  100.000%
- ¦    int64 ¦     10.8 10^9 ¦          ************************ ¦   97.293%
* ¦    int64 ¦      4.0 10^9 ¦                         ********* ¦   36.363%
/ ¦    int64 ¦      0.1 10^9 ¦                                  ¦    1.006%

+ ¦  float32 ¦      1.0 10^9 ¦         ************************* ¦  100.000%
- ¦  float32 ¦      1.0 10^9 ¦          ************************ ¦   98.209%
* ¦  float32 ¦      0.6 10^9 ¦                   *************** ¦   60.568%
/ ¦  float32 ¦      0.2 10^9 ¦                             ***** ¦   23.762%

+ ¦  float64 ¦      5.6 10^9 ¦         ************************* ¦  100.000%
- ¦  float64 ¦      5.6 10^9 ¦         ************************* ¦  100.005%
* ¦  float64 ¦      5.6 10^9 ¦          ************************ ¦   98.618%
/ ¦  float64 ¦      0.2 10^9 ¦                                 * ¦    4.234%
```