Stack and heap memory allocation and access

C++

The only language option for now where we can control memory allocation.

First test is array creation and access.
Results are:

```
Stack creation of 1000000 elements took 0.00013 ms
Heap creation of 1000000 elements took 0.015389 ms
Stack access of 1000000 elements took 7.73307 ms
Heap access of 1000000 elements took 5.57531 ms
```

From run to run both stack and heap may vary, but allocation of the array in heap is always ~1000 times slower than in stack.

Next test is memory allocation of structures, *struct* statically and *class* both dynamically and statically, results are:

```
Struct creation of took 0.000137 ms
Class creation took 0.014003 ms
Class static creation took 0.000128 ms
Class access of 100000000 times took 206.208 ms
Class static access of 100000000 times took 144.777 ms
Struct access of 100000000 times took 134.326 ms
```

Again, access execution times may change from run to run and it is hard to conclude which memory location is actually more efficient.

Note: in C++ struct and class internally are identical and being both allocated in stack does not really differ in creation and access time, which is therefore expected.

Swift

In swift, it is possible to create two entities in different places (stack/heap) by using enum/struct and classes.
Enum/structs are defined in stack and largely used in Swift apps because of the possibility of extension, so the class functionality may be achieved.

Unfortunately, the access test didn’t show any notable difference in performance, therefore in simple cases there is almost no difference which one to use from execution time point of view.