/INFOMOV/

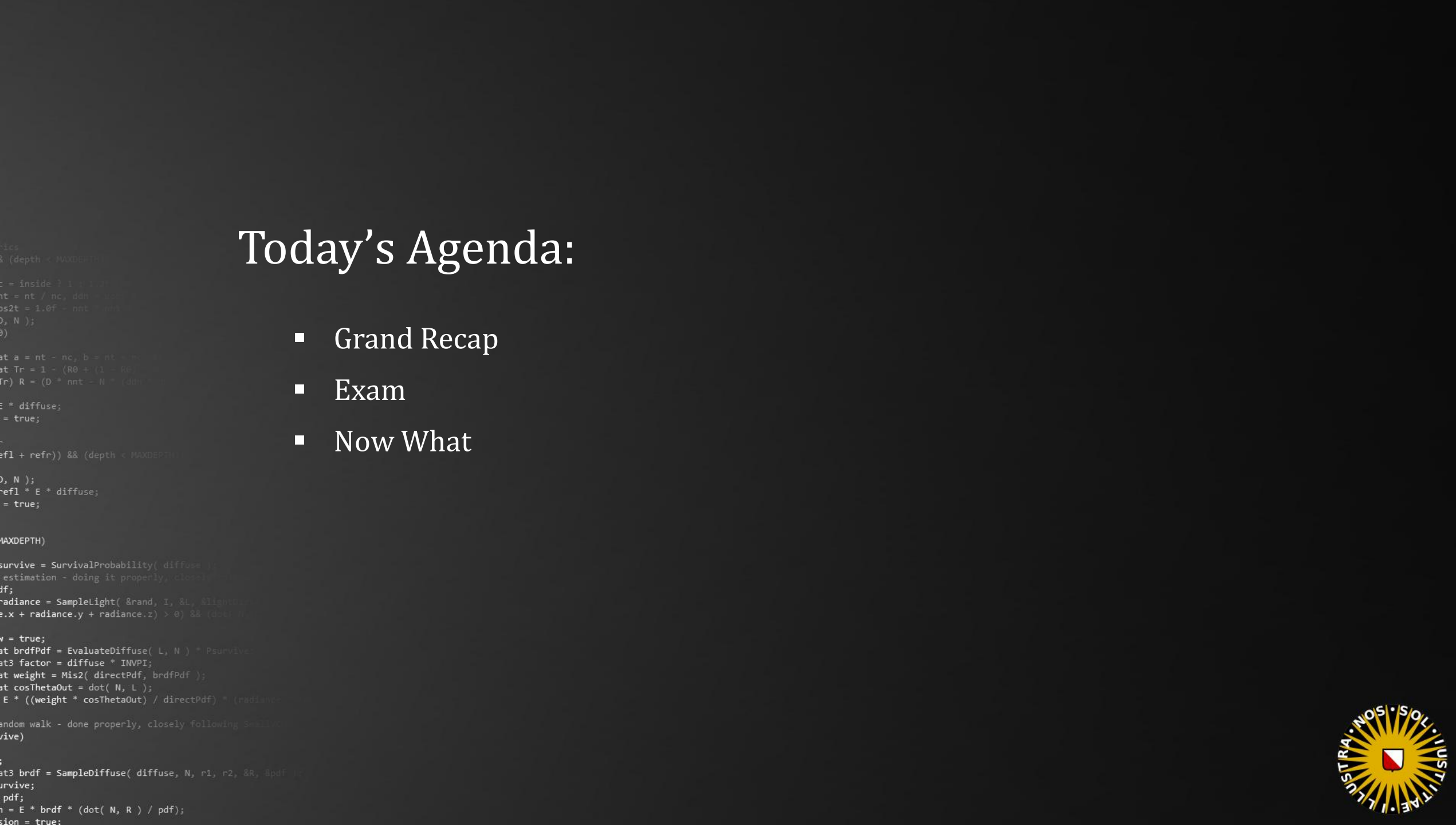
Optimization & Vectorization

J. Bikker - Sep-Nov 2019 - Lecture 14: “Grand Recap”

Welcome!

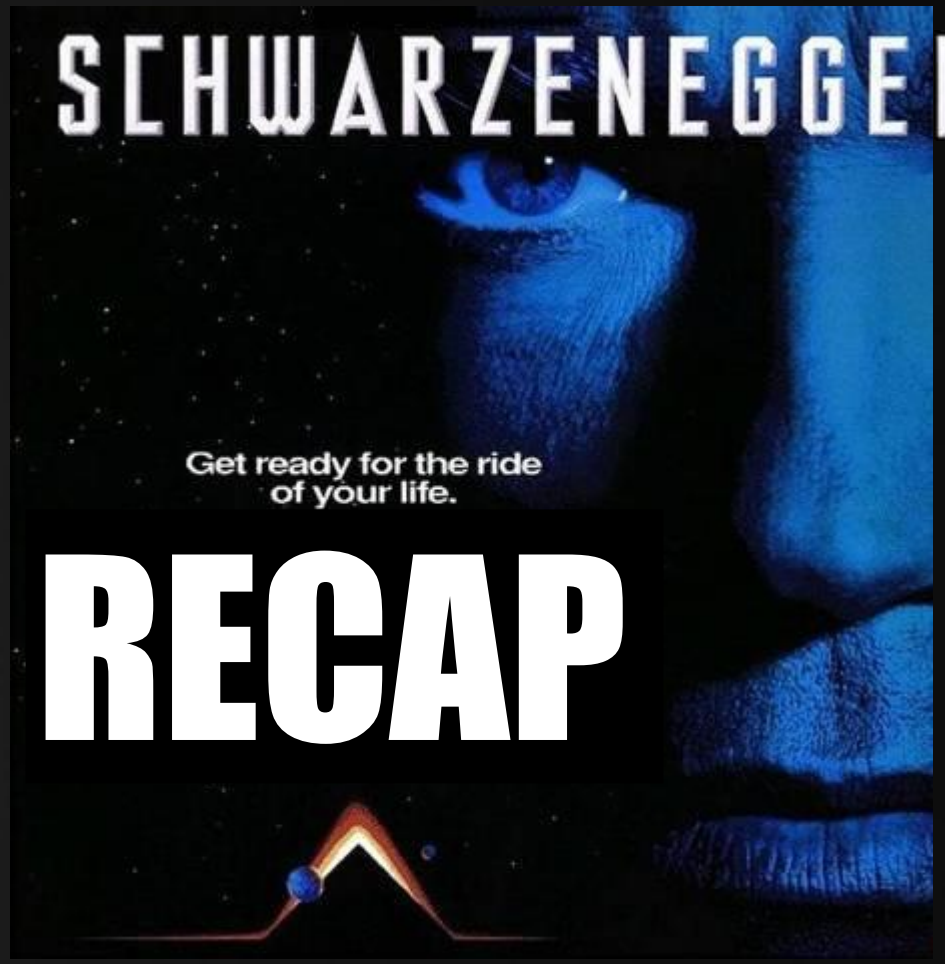
Today’s Agenda:

- Grand Recap
- Exam
- Now What



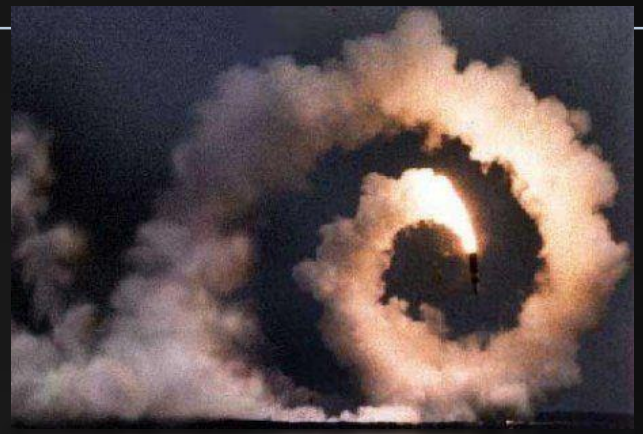
Today's Agenda:

- Grand Recap
- Exam
- Now What

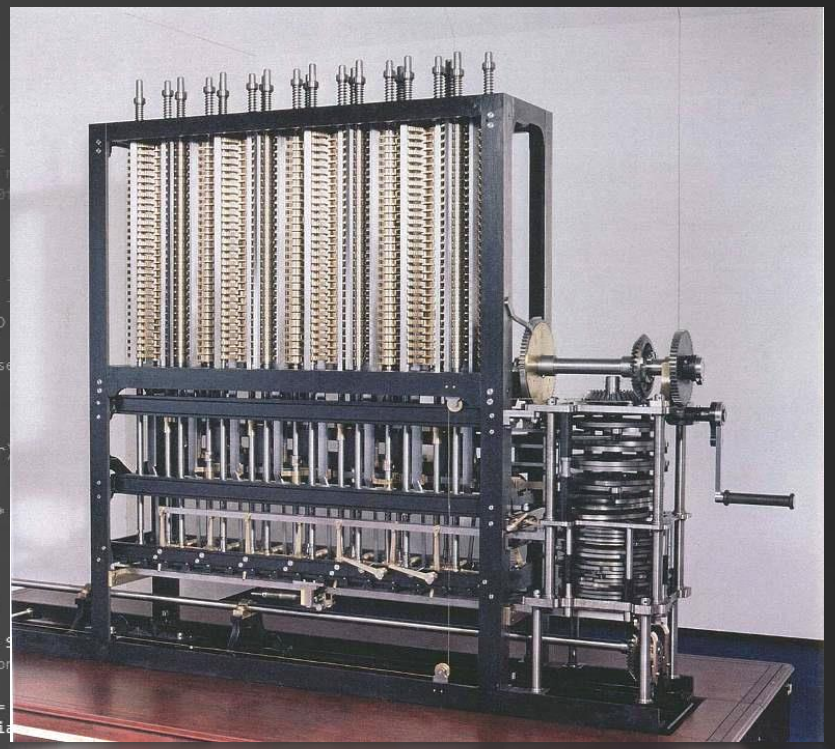


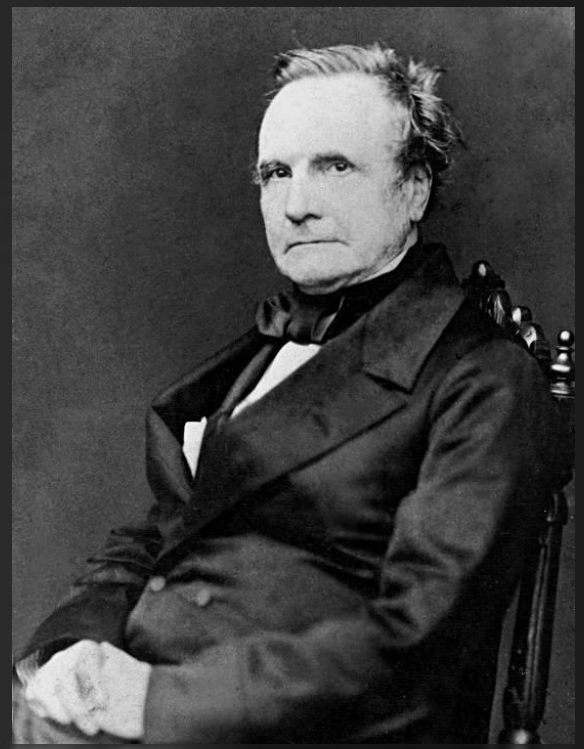
TOTAL RECAP

Recap



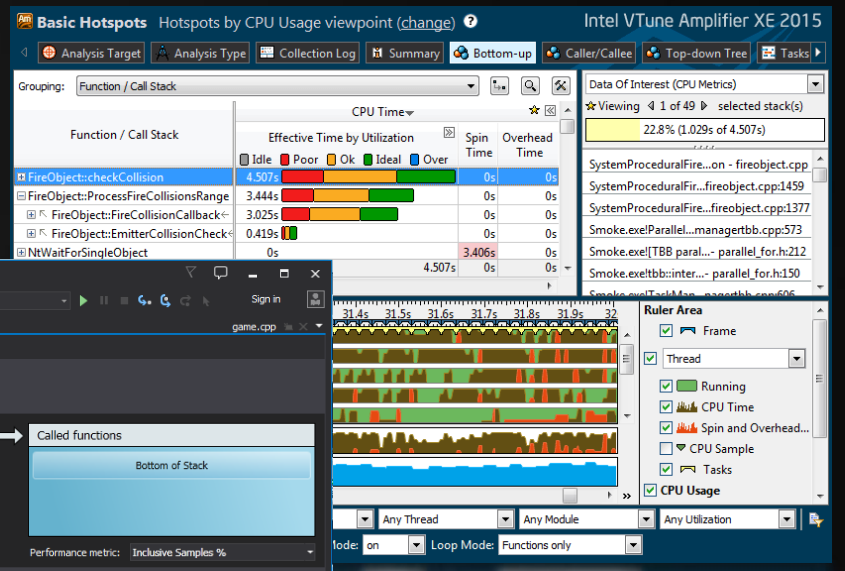
Recap – lecture 1

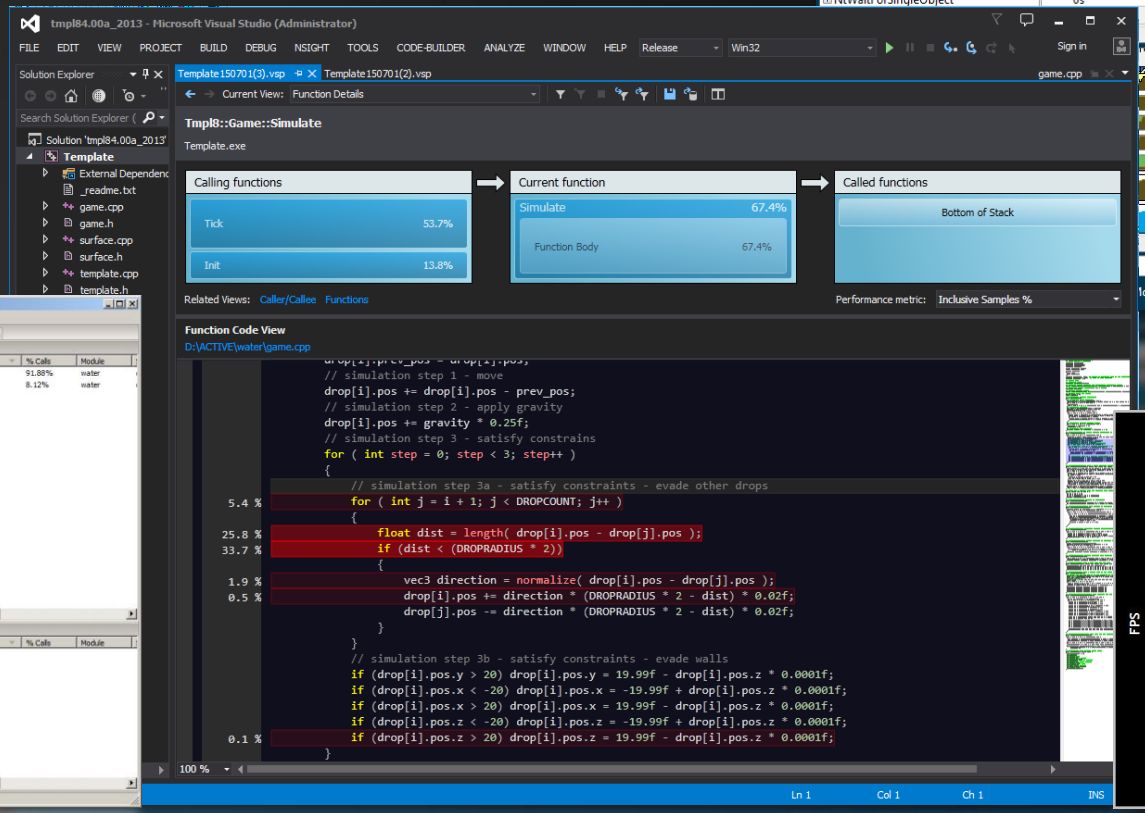


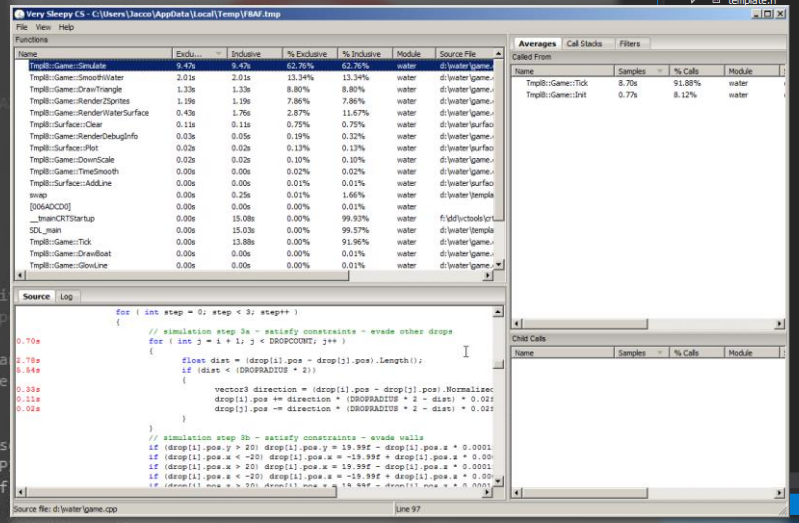


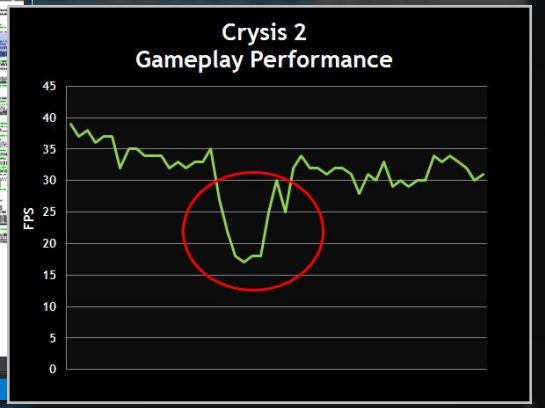
- Profiling
- High Level
- Basic Low Level
- **Cache & Memory**
- Data-centric
- Compilers
- Fixed-point Arithmetic
- CPU architecture
- **SIMD**
- GPGPU

Recap – lecture 1

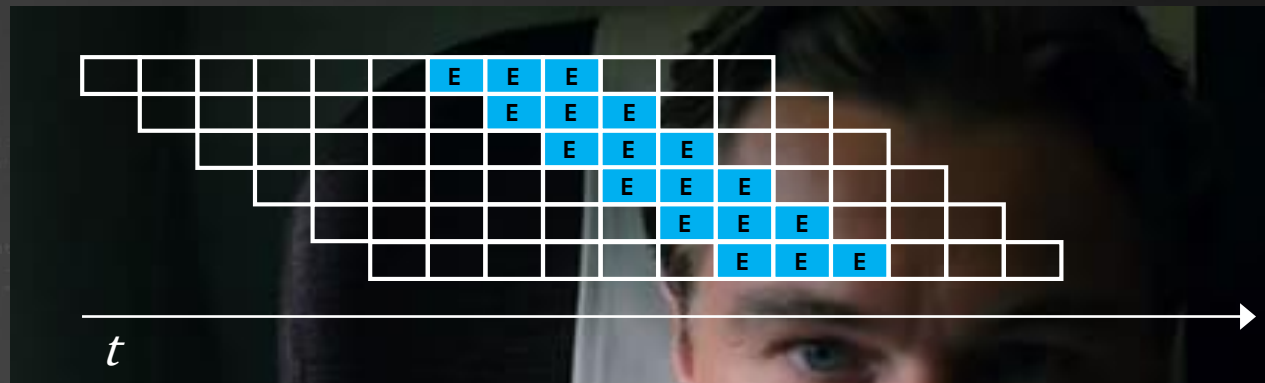








Recap – lecture 2



Red = u4 & (255 << 16);
Green = u4 & (255 << 8);
Blue = u4 & 255;

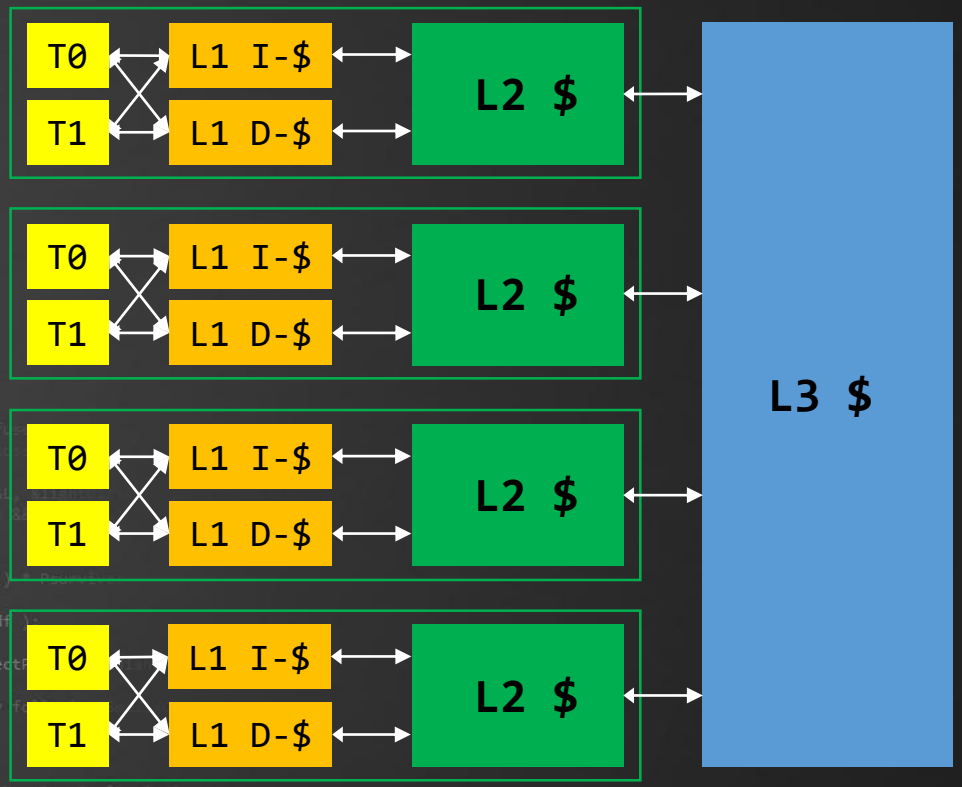

```
fldz
xor ecx, ecx
fld dword ptr ds:[405290h]
mov edx, 28929227h
fld dword ptr ds:[40528Ch]
push esi
mov esi, 0C350h    = 50000
```

```
add ecx, edx
mov eax, 91D2A969h
xor edx, 17737352h
shr ecx, 1
mul eax, edx
fld st(1)
faddp st(3), st
```

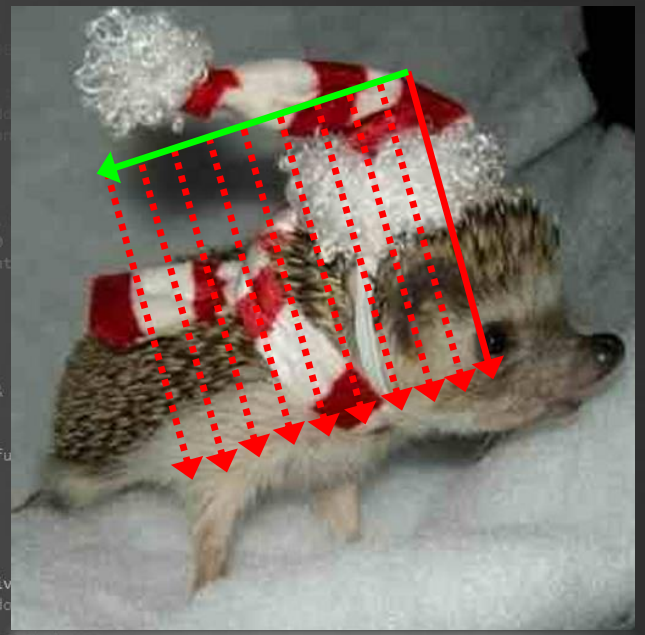
$= \frac{2^{46}}{28763}$ (!!)

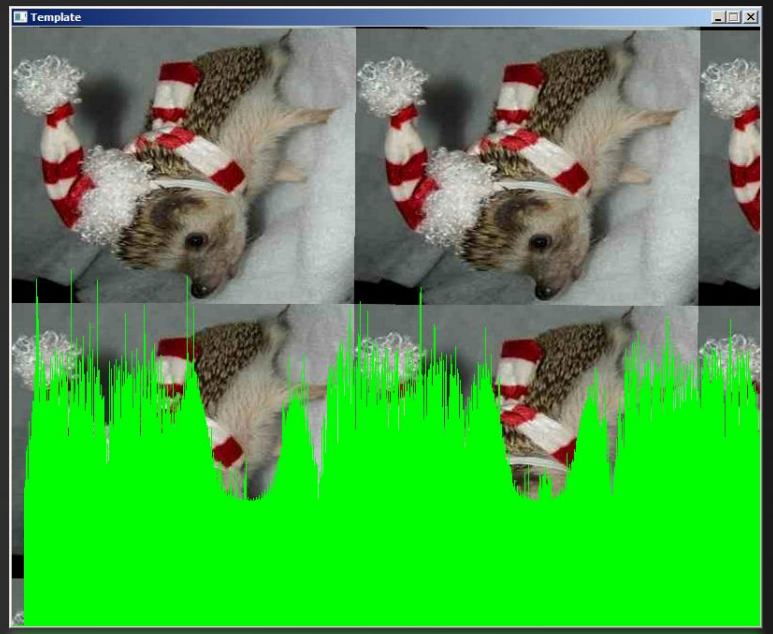
```
mov eax, 91D2A969h
shr edx, 0Eh
add ecx, edx
fmul st(1),st
xor edx, 17737352h
shr ecx, 1
mul eax, edx
shr edx, 0Eh
dec esi
jne tobetimed<0>+1Fh
```

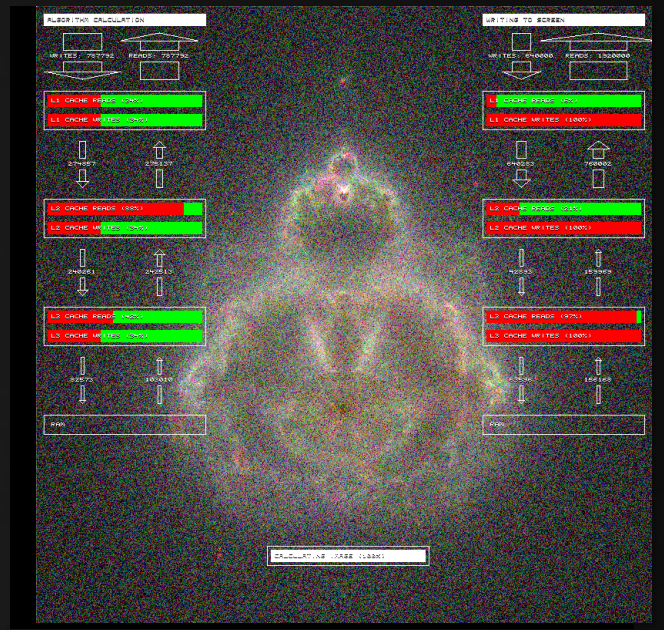
Recap – lecture 3



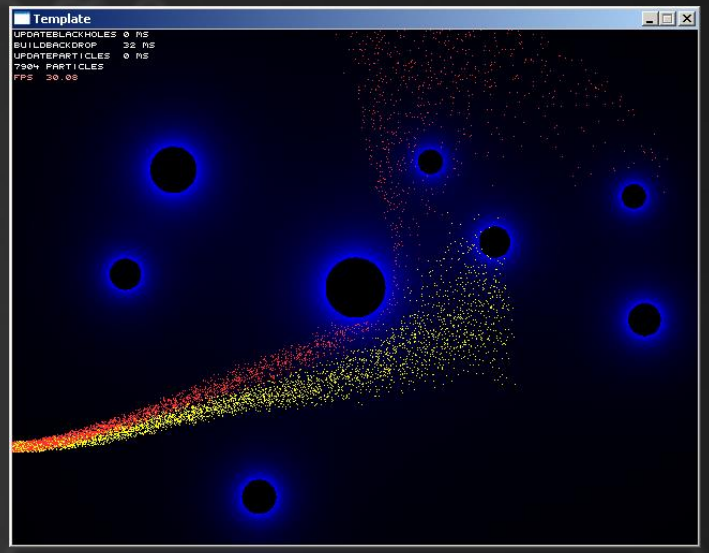
Recap – lecture 4

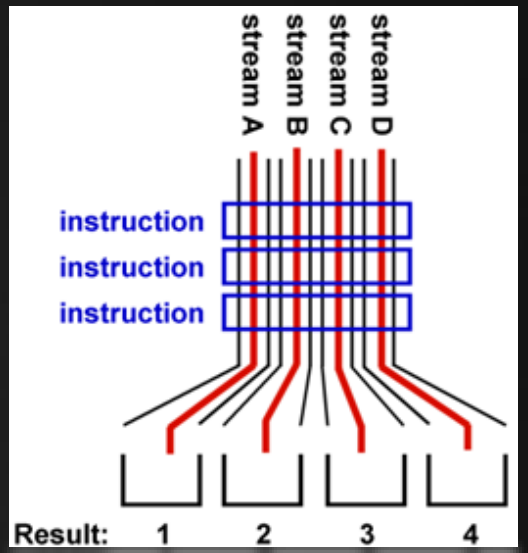






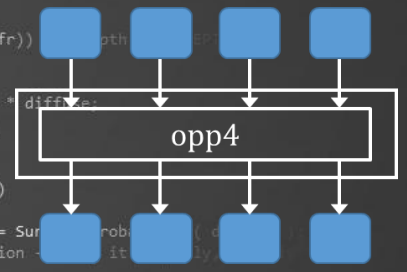
Recap – lecture 5 & 6





Agner says:
“Automatic vectorization is the easiest way of generating SIMD code, and I would recommend to use this method when it works. Automatic vectorization may fail or produce suboptimal code in the following cases:

- when the algorithm is too complex.
- when data has to be re-arranged in order to fit into vectors and it is not obvious to the compiler how to do this or when other parts of the code needs to be changed to handle the re-arranged data.
- when it is not known to the compiler which data sets are bigger or smaller than the vector size.
- when it is not known to the compiler whether the size of a data set is a multiple of the vector size or not.
- when the algorithm involves calls to functions that are defined elsewhere or cannot be inlined and which are not readily available in vector versions.
- when the algorithm involves many branches that are not easily vectorized.
- when floating point operations have to be reordered or transformed and it is not known to the compiler whether these transformations are permissible with respect to precision, overflow, etc.
- when functions are implemented with lookup tables.



AoS

SoA

SIMD Basics

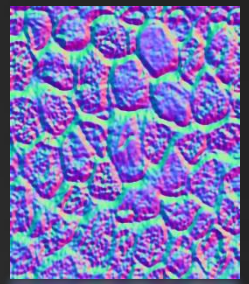
Other instructions:

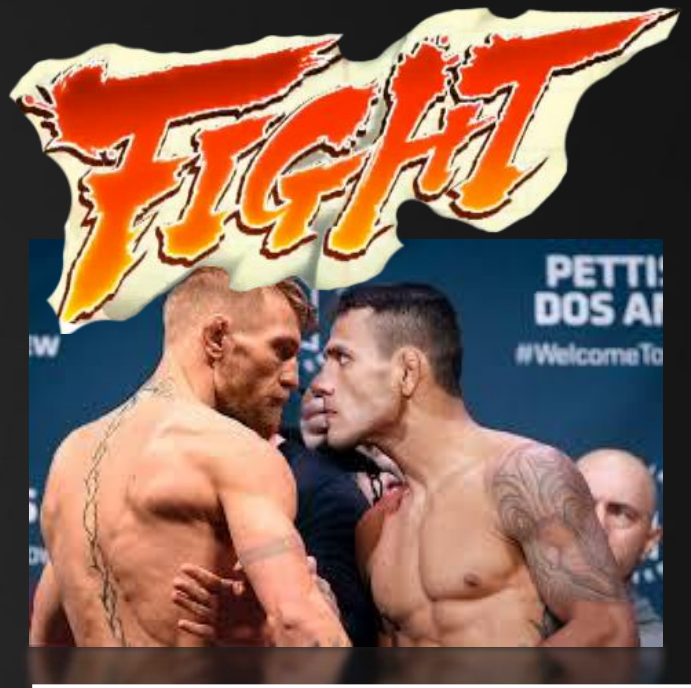
```
__m128 c4 = _mm_div_ps( a4, b4 ); // component-wise division
__m128 d4 = _mm_sqrt_ps( a4 );    // four square roots
__m128 d4 = _mm_rcp_ps( a4 );     // four reciprocals
__m128 d4 = _mm_rsqrt_ps( a4 );   // four reciprocal square roots (!)

__m128 d4 = _mm_max_ps( a4, b4 );
__m128 d4 = _mm_min_ps( a4, b4 );
```

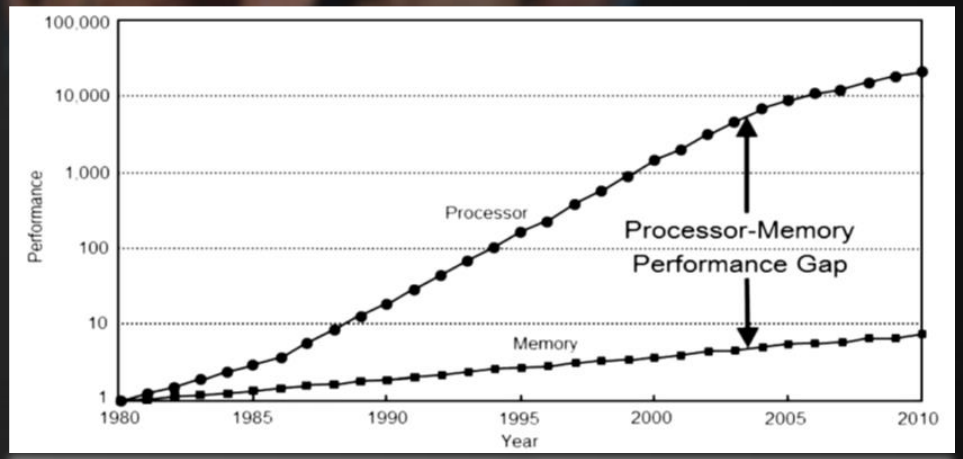
Keep the assembler-like syntax in mind:

```
__m128 d4 = dx4 * dx4 + dy4 * dy4;
```

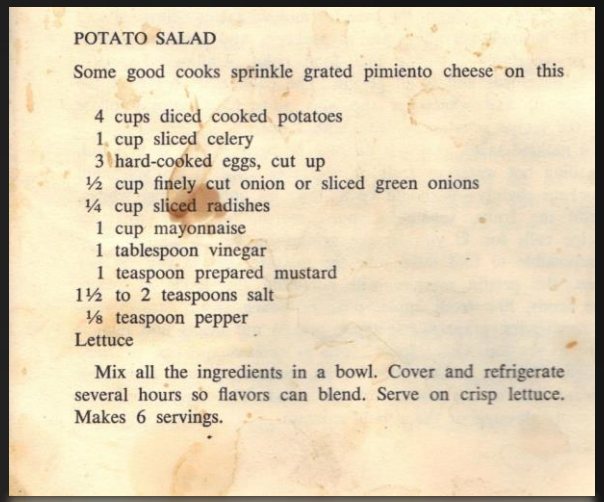
Recap – lecture 7







Recap – lecture 8

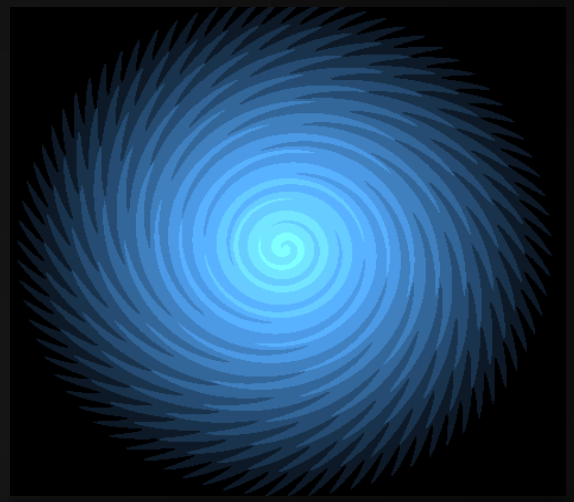


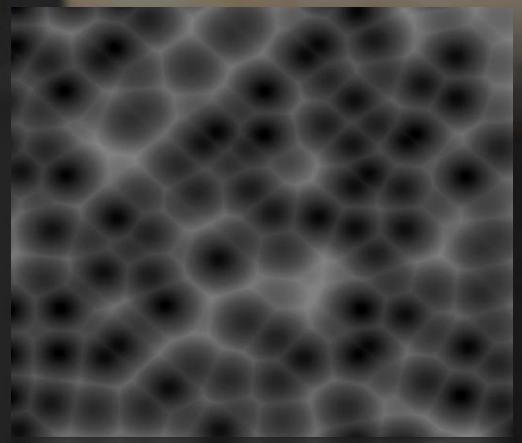
POTATO SALAD

Some good cooks sprinkle grated pimiento cheese on this

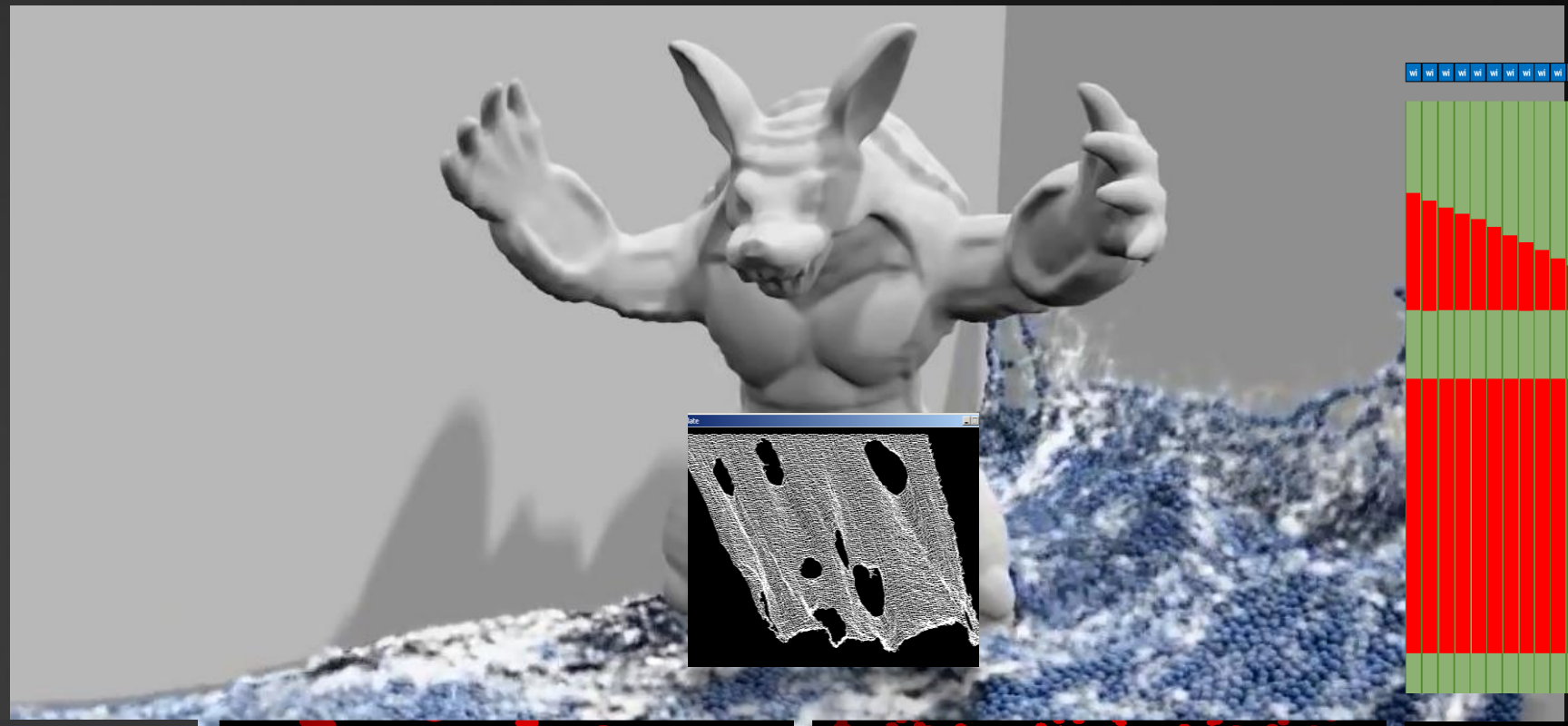
4 cups diced cooked potatoes
1 cup sliced celery
3 hard-cooked eggs, cut up
½ cup finely cut onion or sliced green onions
¼ cup sliced radishes
1 cup mayonnaise
1 tablespoon vinegar
1 teaspoon prepared mustard
1½ to 2 teaspoons salt
⅛ teaspoon pepper
Lettuce

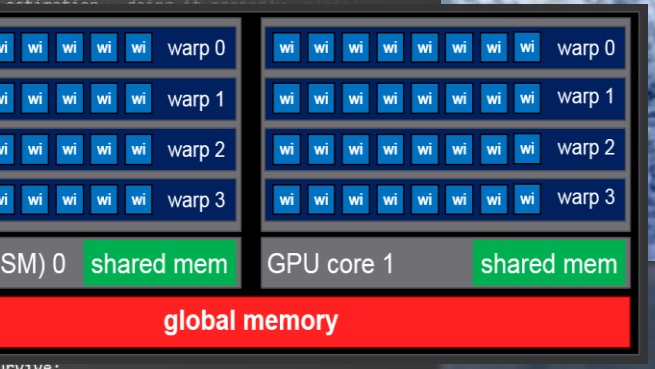
Mix all the ingredients in a bowl. Cover and refrigerate several hours so flavors can blend. Serve on crisp lettuce. Makes 6 servings.

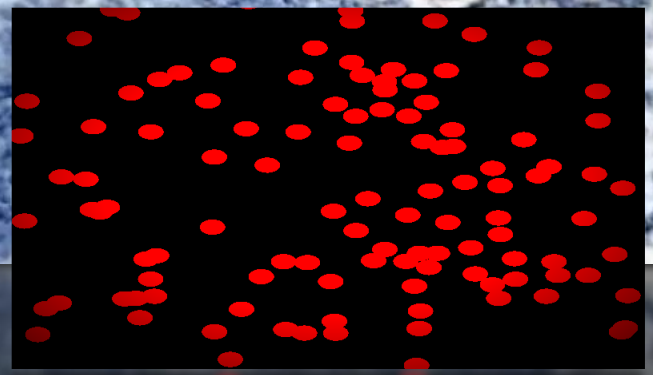


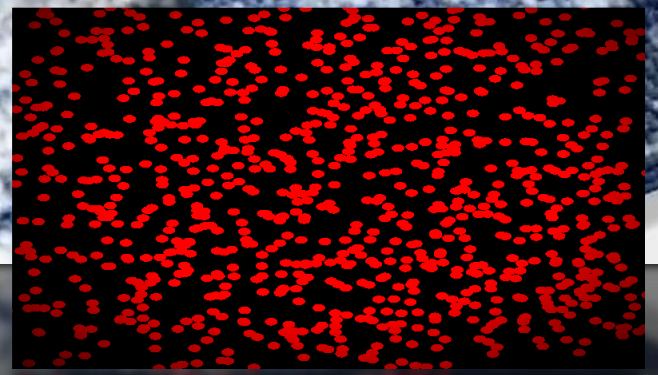


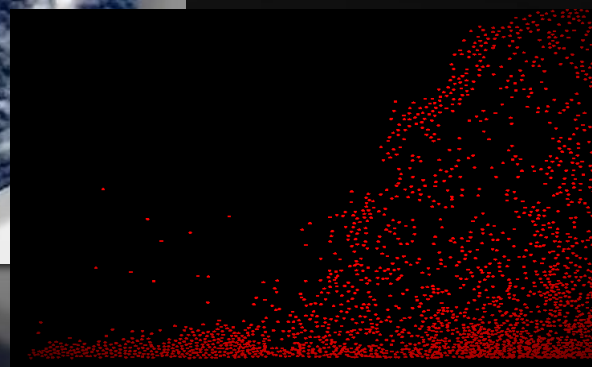
Recap – lecture 9 & 10



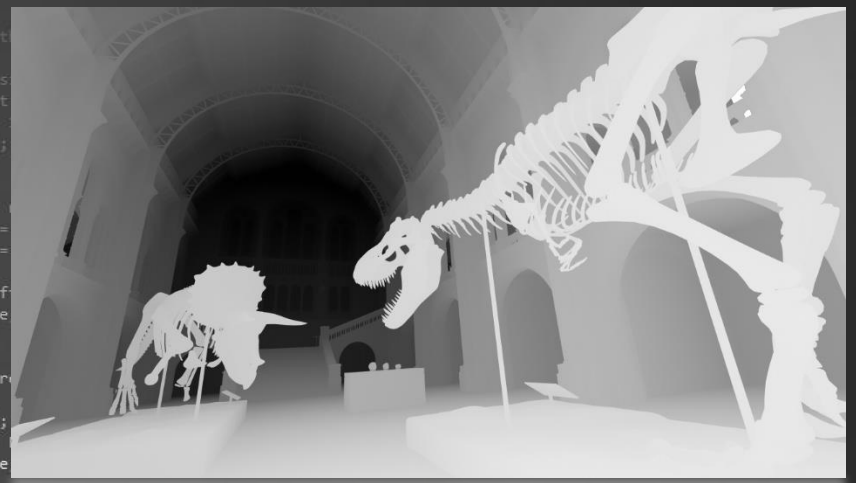


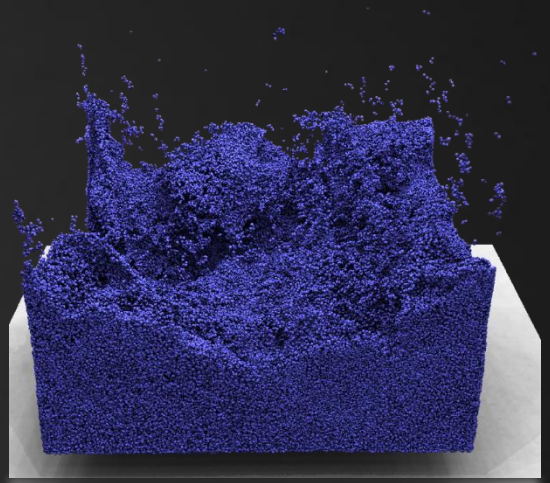


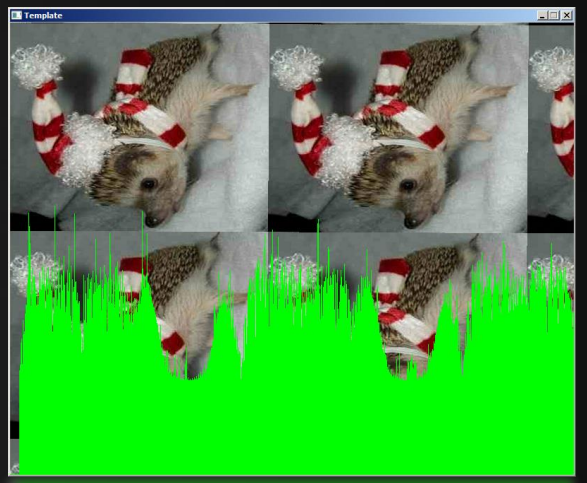


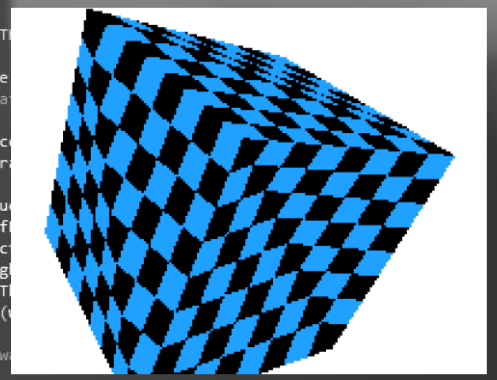


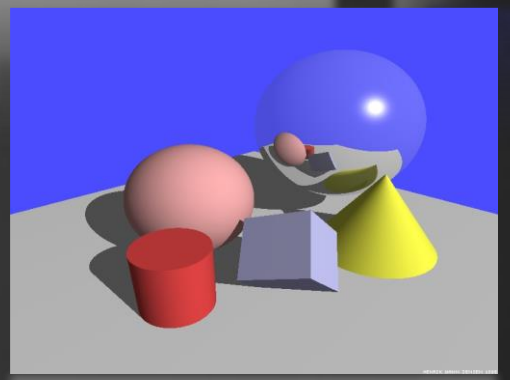
Recap – lecture 11



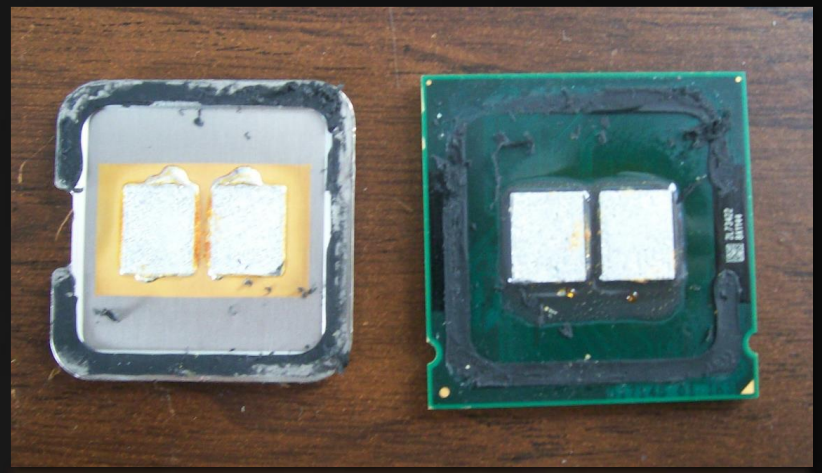


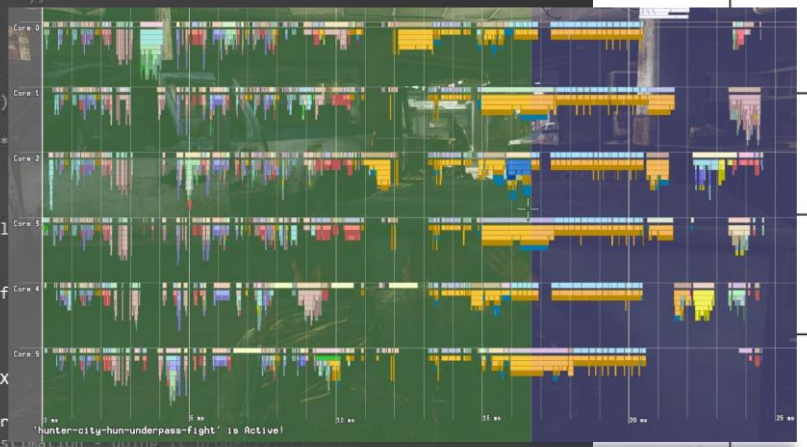




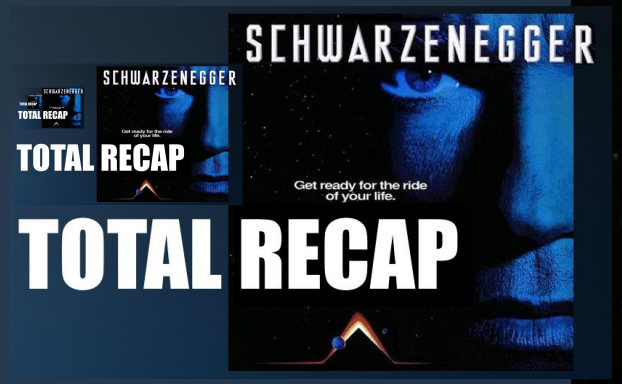


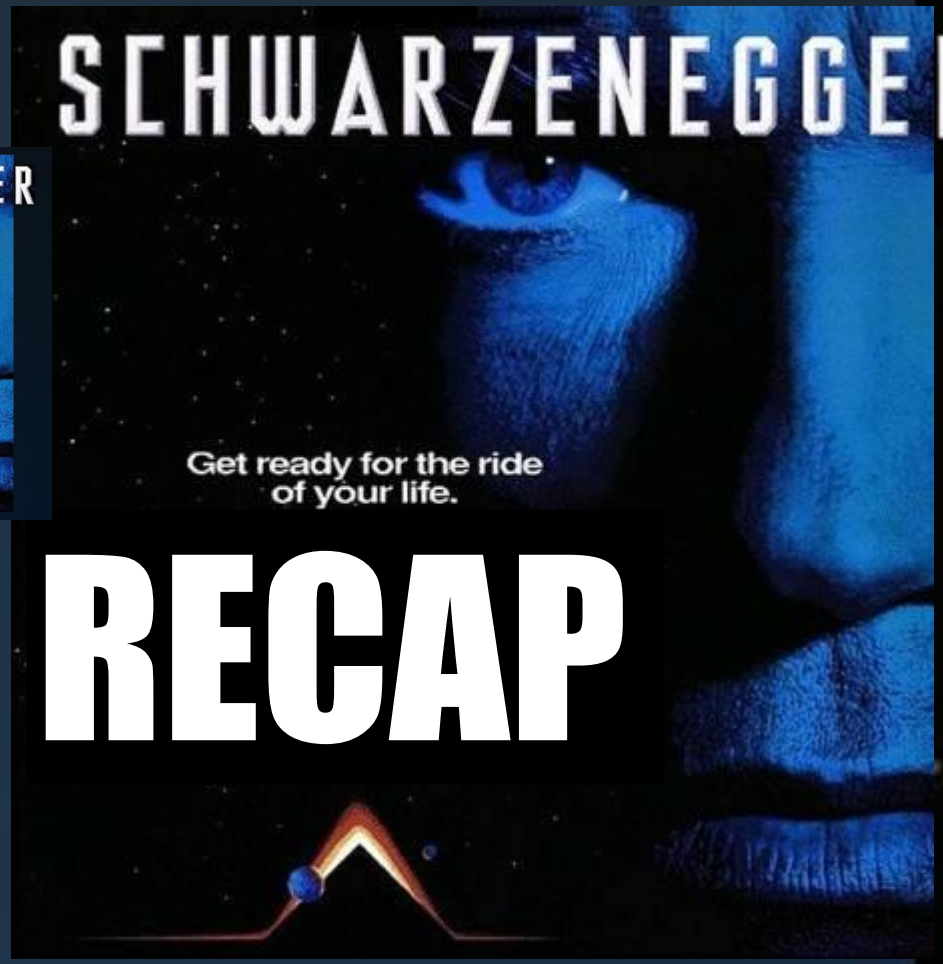
Recap – lecture 13





Recap – Lecture 14





TOTAL RECAP

Recap

“Dear Charles,

At the end of this course:

You will know how to speed up critical code by a factor 2x to 10x (and more).

- You will be able to do this to virtually any program*.
- Your understanding of higher level optimization approaches will increase.
- You will be able to apply these principles to new / alien hardware.
- You will have a more intimate relationship with your computer.

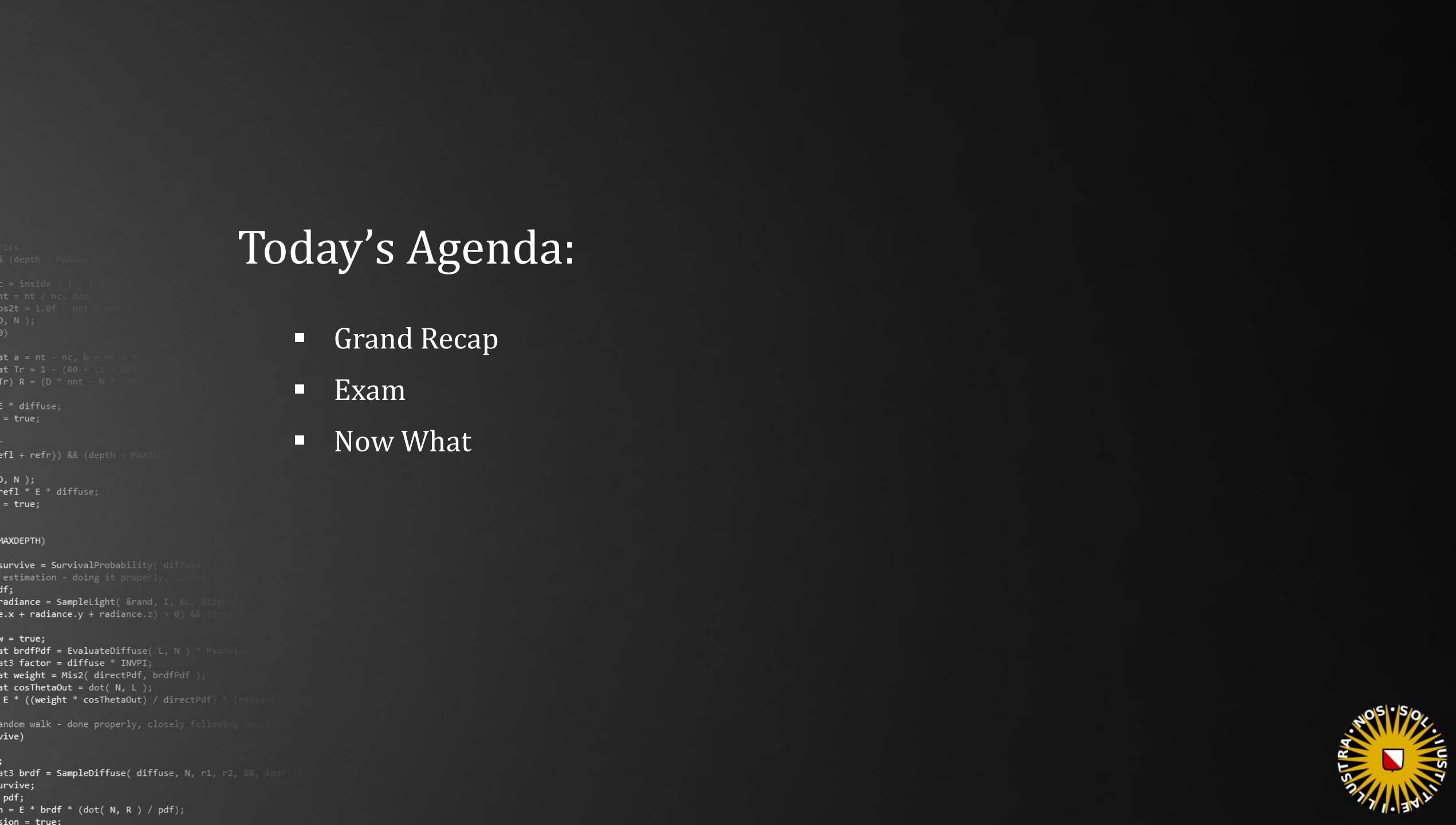
In other words:

We will talk a lot about the ‘C’ in O(N).

* disclaimer: ‘that has not been optimized by an expert’.

Today’s Agenda:

- Grand Recap
- Exam
- Now What



Exam

What to Study

1. Slides
2. Literature on the website and in the slides:
   - Modern Microprocessors: a 90 minute guide, see lecture 2 slides or click here
   - What Every Programmer Should Know About Memory (just the yellow bits)
   - Gallery of Processor Cache Effects (link)
   - Game Programming Patterns - Data Locality
   - Data-Oriented Design (Or Why You Might Be Shooting Yourself in the Foot With OOP)
   - The Neglected Art of Fixed Point Arithmetic
   - Cache-oblivious Algorithms and Data Structures (just the yellow bits)
   - A Survey of General-Purpose Computation on Graphics Hardware
3. 2016/2017/2018 exams
4. Skills you picked up with the practical assignments

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

CPUs and GPUs have fundamentally different core strategies for dealing with latencies such as memory access time. What are these strategies?

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

Why is the theoretical peak performance of a GPU typically much higher than that of a CPU?

Exam

Example Questions

What is DMA?

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

Explain the concept of streaming processing.

Exam

Example Questions

What or who is NUMA?

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

Explain what false sharing is.

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

How does a GPU handle conditional code?

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

Why does OpenCL have a native_sqrt as well as an sqrtf?

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

Do modern systems still use SRAM? Why / why not?

Exam

You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

How many bits are needed for a 128KB 8-way set associative cache, assuming a cache line size of 128 bytes?

Exam

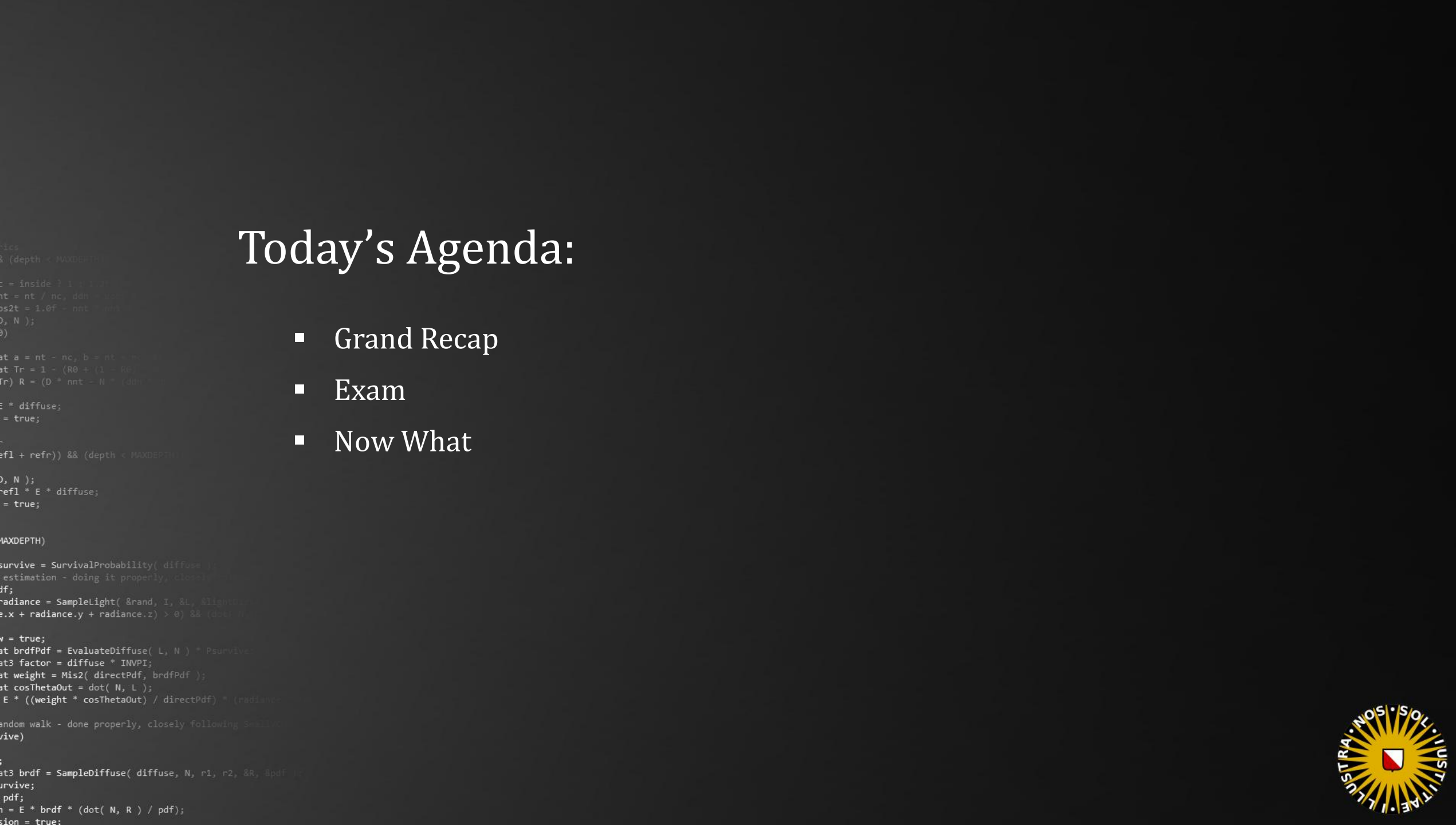
You may bring a dictionary to the exam.
You may answer in Dutch, if you wish.
You may **not** bring notes to the exam.
You may bring pizza to the exam.

Example Questions

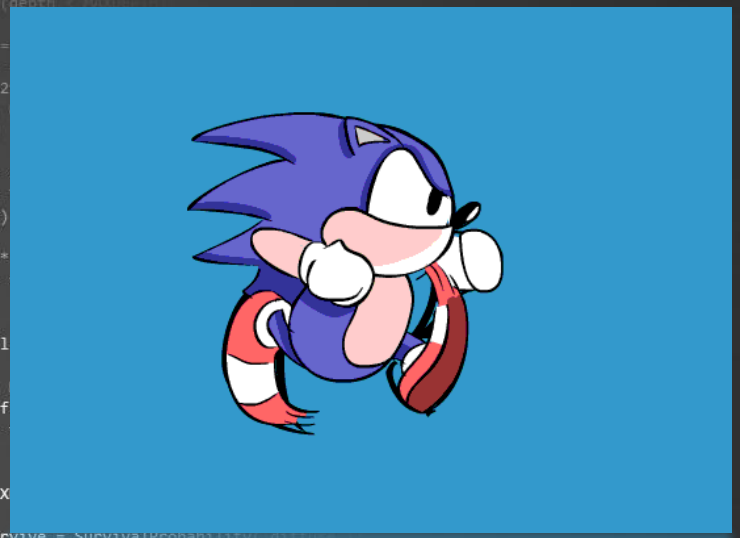
Is self-modifying code possible on a modern processor? Under what conditions?

Today’s Agenda:

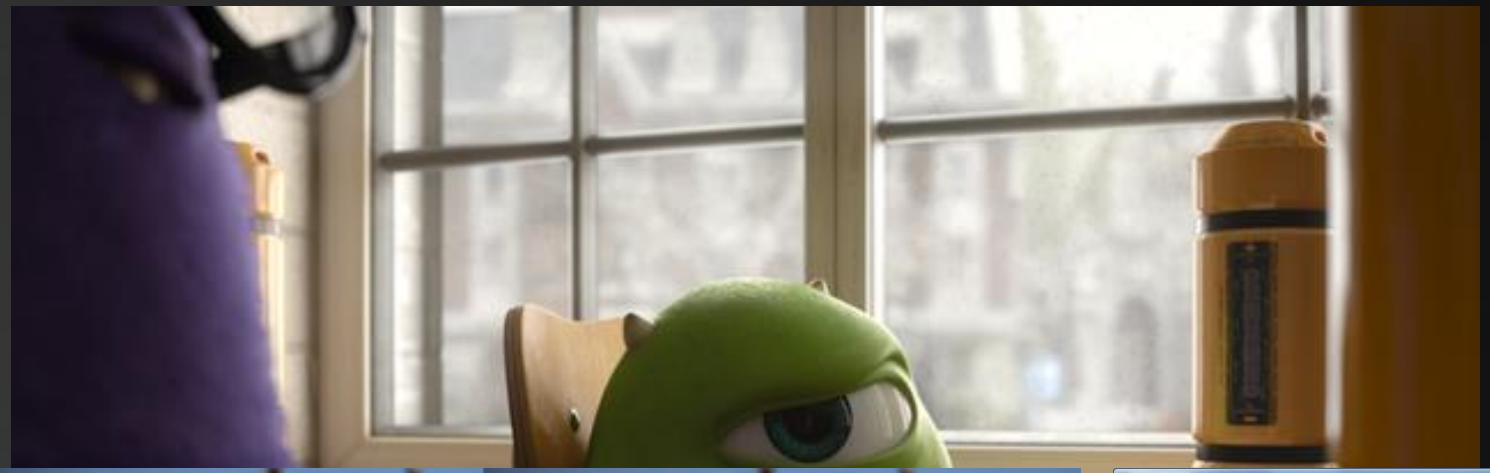
- Grand Recap
- Exam
- Now What

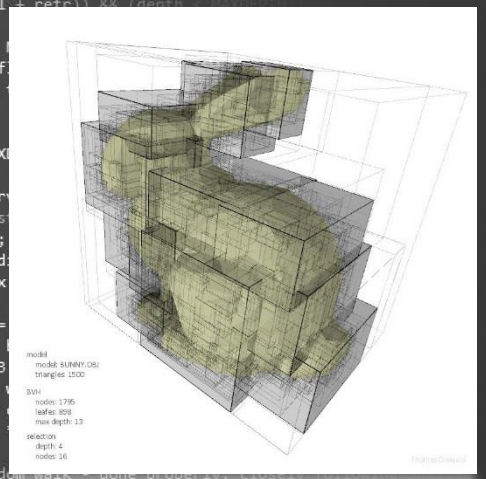


Now What



Now What





Now What

Now What


COMPUTER PROGRAMMER
What my friends think I do
What my mom thinks I do
What society thinks I do
What my spouse thinks I do
What I think I do
What I actually do

/INFOMOV2019/