# Lecture 8
# 1D Arrays

and other disasters

# Test is coming

## 5'th December

Know:

- Data types
- Functions
- I/O operations
- Branching (if, switch)
- Loops

Have:

- A Pen
- Student ID

**Do not** have:

- Notes
- Any electronic devices

Name and Last name: XXXXXXXXXXXXXXXXXX Index Number: | 1 | 2 | 3 | 4 | 5 | 6 |

Answer questions in the spaces provided on the question sheets. All parts of questions 1,2 and 3 **MUST** be answered to receive any points. Write your name in **printed** letters. **Unreadable or unformatted answers will not be checked!**

1. Correct compilation errors in the following code sections. In the boxes below write the line number you are correcting and the corrected code that should be put in that line only. If you think the code will compile write **Will Compile** in a box. Assume used variables, functions and includes have been declared.

```
1for(int i=0, i<10, ++i){
2 printf("%d\n", i);
3}
```

a)

```
1do{
2 printf("%d\n", i);
3 ++i;
4}while(i<10);
```

b)

```
1for(int i=0; i<0; ++i);
2{
3 printf("%d\n", i);
4}
```

c)

2. Write the output of the following code sections in the boxes. Assume used variables, functions and includes have been declared.

```
1int i=0;
2for(; i<10; ++i){
3 break;
4}
5printf("%d\n", i);
```

a)

```
1int a=4;
2if(a%3) a=5;
3printf("%d\n", a);
```

b)

```
1int a=0;
2for(int i=0; i<10; ++i){
3 a=i;
4}
5printf("%d\n", a);
```

c)

3. Write a **complete program** printing odd numbers from 0 to 1000, but not those dividable by 9. The program must fit in the boxes below:

4. Write a **function** of an appropriate type that for 3 double arguments $a$, $b$, $c$ calculates and returns the result of a following formula: $\sum_{i=1}^{40} a\, e^{\frac{i}{i+1}} + \frac{1}{b}$

# Today
## 2D static arrays

Today:

- File I/O - continue
- A program that preperas points for plotting a function
- A program that reads data from a file, and manipulates them.
- A program that generates random numbers and stores them in a file.
- A program that reads a file, and calculates an average
- A program that reads a file, sorts it and stores back

# Files

FILE structure to handle files:

```
FILE *fp;
```

To open a file use *fopen()*:

```
FILE *fopen(const char *filename, const char *mode);
//e.g.:
fp=fopen("c:\\test.txt", "r");
```

To close a file use *fclose()*:

```
int fclose(FILE *a_file);
//e.g.:
fclose(fp);
```

# Files

## fopen modes

Depending on what we require the file to:

- r - open for reading
- w - open for writing (file need not exist)
- a - open for appending (file need not exist)
- r+ - open for reading and writing, start at beginning
- w+ - open for reading and writing (overwrite file)
- a+ - open for reading and writing (append if file exists)

# Files

## Reading and writing with fprintf, fscanf

Printing to file:

```
FILE *fp;
fp=fopen("c:\\test.txt", "w");
fprintf(fp, "Testing...\n");

...
fclose(fp);
```

Reading from file:

```
FILE *fp;
fp=fopen("c:\\test.txt", "w");
int a;
fcanf(fp, "%d", &a);
...
fclose(fp);
```

# Examples

Use static arrays only.

1. Write a program that writes to a file coordinates to plot $f(x) = sin(x)$ for a range $< 0, 2\pi >$
2. Write program that reads points coordinates from a file and decides if those are in a circle of radius 1.
3. Write a program that generates N random numbers and stores them to a file.
4. Write a program that reads a data file, calculates an average value and finds the number of elements above, and below that average.
5. Write a program that reads values from a file, sorts them and stores them to a new file.
6. Example test questions