Strings

- What is a text?
- A string of characters
- String usually ends with \0
- Incredibly unsafe, almost anything will be unsafe and let hackers into your code!
  Jokes aside, let’s go on a journey

Prehistoric ages

```c
#include <stdio.h>

int main () {
   char str[10];
   printf("Enter a string : ");
   gets(str);
   printf("You entered: %s“, str);
   return(0);
}
```

```
~/Desktop took 4s
→ ./a.out
warning: this program uses gets(), which is unsafe.
Enter a string : hello
You entered: hello%
```

What could go wrong?

- gets starts at the buffer start and puts everything it reads into that buffer

- What if the buffer is too small?
  Bufferoverflow occurs

- Many terrible things can happen

- Exploits are created this way

```c
int main () {
    char str[10];
    printf("Enter a string : ");
    fgets(str, 10, stdin);
    printf("You entered: %s", str);
    return(0);
}
```

```
~/Desktop
↓9% → ./a.out
Enter a string : hello
You entered: hello

~/Desktop took 2s
↓9% → ./a.out
Enter a string : hello world
You entered: hello wor%
```

It isn’t that simple!

```c
int main () {
   char *hello = "Hello ";
   char *world = "World";
   hello = hello + world;
   printf("You entered: %s", hello);
   return(0);
}
```

You can’t really add two pointers

```c
char *hello = "Hello ";
char *world = "World";
hello = world;
```

What happens to „Hello “?

There are no Strings in C
Everything is an array of chars

How do you append a char array to another?

- 1) Have a char array that can store both arrays
- 2) Find the end of the first array
- 3) Append each char after that end point one by one
- 4) Add a \0 to mark the end

That sucks!

- Yes it does, Timmy!
- #include <string.h> for some functions that are able to help you
- strcpy(s1, s2) copies s2 into s1
- strcat(s1, s2) appends s2 after s1
- strlen(s1) returns the length of s1

Why does this crash? // better example

```
char hello[6] = "Hello ";
char world[6] = "World ";
strcpy(hello, world);
printf("You entered: %s", hello);
```

\0 is appended, thus it needs 7 chars

Command Line parameters

- There are two versions of the main function
- int main(void)
  int main(int argc, char **argv)

- argc indicates how many parameters are passed

- argv is a pointer to a pointer of a char

Pointer to Pointer to char? //cmd line parameters

- argv[0] contains a pointer to a char array
- argv[1] contains a pointer to another char array
- As each char array may have a different length, a pointer to an array of char arrays might over/underjump when accessing next element
- Therefore some pointers are saved, which point to the parameters

```c
int main (int argc, char **argv) {
  for(int i = 0; i < argc; i++){
    printf("%s\n", argv[i]);
  }
  return(0);
}
```

```
~/Desktop
→ ./a.out hello world
./a.out
hello
world
```

```c
int main (int argc, char **argv) {
  for(int i = 0; i < argc + 10; i++){
    printf("%s\n", argv[i]);
  }
  return(0);
}
```

```
→ ./a.out hello world
./a.out
hello
world
(null)
TMPDIR=/var/folders/41/jc8sjvmd52q2cz20fp8wnqtr0000gn/T/
__CF_USER_TEXT_ENCODING=0x1F5:0x0:0x3
SHELL=/bin/zsh
HOME=/Users/peterzdankin
Apple_PubSub_Socket_Render=/private/tmp/com.apple.launchd.W35tKYFeDb/
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.i2dQoWNdFq/Listeners
PATH=/Users/peterzdankin/.cargo/bin:/usr/local/bin:/usr/bin:/bin:/usr
ions/Wireshark.app/Contents/MacOS:/Users/peterzdankin/.cargo/bin
LOGNAME=peterzdankin
XPC_SERVICE_NAME=0
```

Exercises

- Write a function, that uses pointers to modify our awesome fortnite functions from last week
- Write a function that gets a „string“ and returns the string in uppercase
- Write a function that checks whether an input „string“ is a valid IPV4 Address (0-255.0-255.0-255.0-255)
- Find out what is the biggest array you are allowed to use in a program

Even more Exercises

- Write a safe strcat function, that gets two char* and the length of the first char* and appends the second to the first. Remember \0

- You get a string of four rows of characters, write a function that does a horizontal flip and write a function that does a vertical flip (e.g. h: „ab\ncd“ -> „ba\ndc“)

- Try returning a stack „string“ and print it out, what happens?