Lab-04

BE5B99CPL – C Programming Language

HW assignment

- HW 1 Reminder, deadline this evening!
- HW 2 Reminder, deadline next week!
- HW 3

Static Memory - get user's name

```
char name[999];

printf("Enter your name:\n");

scanf("%s", name);
```

Why is it bad?
Why do so many people do it?

Dynamic Memory - get user's name

scanf requires that it is given a pre-allocated buffer ahead of time

Risk of memory leak - writing past the end of the buffer

What can we do?

- limit number of characters to be read scanf("%998s")
- **dynamically allocate memory**

Valgrind

- Many programmers forget to *free* their memory after using it
- You can use *valgrind* for memory leak detection
- It can detect forgotten allocated memory and the exact location of out-of-bound access
- Can report using uninitialized variables/memory
- For better output, compile your program with *-g* flag

valgrind ./your-program

Pointers

- Assume the following declaration:

```
int a = 1;
double d = 1.7;
int *iptr;
double *dptr;
```

Which of the assignments are incorrect and why?

```
iptr = &a;
dptr = &a;
dptr = iptr;
*iptr = d;
*dptr = *iptr;
```

Strings

- Explain the difference between declarations. Are they correct?

  char *string = "ABC";

  char *letter = 'A';

  const char *const letter2 = 'A';

  const char *const letter3 = "A";

Strings

- Explain the difference between declarations:

  char a[]="string";

  char *p="string";

Dynamic Memory - get user's name

Task 1: Write a program which allows the user to enter "infinite" amounts of text, and will keep appending it to a string (char array)

When they press enter, it will output the current string, and if they enter a certain character it will close.

Use the getchar() function to get letters from the command line.

You will need to use malloc/realloc to control the memory.

Don't forget to use free() at the end to free the memory.

Remember to use valgrind and "*-Wall -Werror -pedantic -std=c99 -O2*" to write good code!

string.h contains many helper functions

Library contains many functions for dealing with strings (arrays of characters)

For example, strcmp() will compare two strings, and tell you if they are the same, but it is case-sensitive

Task 2:

- Update the previous code to read a word from the stdin
- Implement your own version of the strcmp() function
- Hard-code your name into your program, and ask the user to input words
- If it matches your name, print a message!
- Can you make it case insensitive?