CENG328
Operating Systems

Laboratory III
C Review II & GNU Debugger

1. C Review II

- Structures; a set of more variables grouped together for convenient handling (code10).

```
#include <stdio.h>
#define PI 3.14

typedef struct point
{
    int x;
    int y;
} Point;

typedef struct circle
{
    Point center;
    int radius;
} Circle;

double get_area(Circle *c);
// ...
```

1. C Review II

```
// ...
int main(int argc, char *argv[])

{
    Point p;
    p.x = 3;
    p.y = 5;
    Circle *circle = (Circle *)malloc(sizeof(*circle));
    circle->center = p;
    circle->radius = 2;
    printf("center: (%d, %d), radius: %d)\nArea: %f\n",
        circle->center.x,circle->center.y,circle->radius,
        get_area(circle));
    return 0;
}

double get_area(Circle *c)
{
    return 2*PI*c->radius;
}
```

- Analyze the code.
- Execute the code. What is the output and why?
- Exercise: Modify the code to;
    - add structure for a cylinder,
    - print out the volume of the cylinder.

1. C Review II

- File Input and Output; examples on how to read from a file and write to a file.
  - code11.c and code12.c.
  - Analyze the code11.c and the output, do not forget to retrieve the file datafile.
  - Analyze the code12.c and the output, what is stored into file "output"?

2. Using GNU Debugger

- GNU Debugger (gdb) is a powerful terminal debugger for Unix-like systems. GDB offers tracing and altering how computer programs are working. Below are some commands:
  - **run**: Used for starting your application under gdb.
  - **step**: Step-by-step program execution.
  - **print <expression>**: Displays the current state of given expression.
  - **break <function>** — **<line>**: Inserts a breakpoint at the given line or function.
  - **clear <function>** — **<line>**: Removes the breakpoint at the given line or function. If no parameters are passed, all breakpoints will be removed.
  - **list**: Prints lines from source code.

2. Using GNU Debugger

```
$ gcc -O0 -g -o code1 code1.c
$ gdb code1
GNU gdb (GDB) 7.2-ubuntu
Copyright (C) 2010 Free Software Foundation,
Inc.
License GPLv3+: GNU GPL version 3 or later
<http://gnu.org/licenses/gpl.html>
This is free software: you are free to change
and redistribute it.
There is NO WARRANTY, to the extent permitted by
law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/ceng328/code1...done.
(gdb) list
1       #include <stdio.h>
2
3       int
4       main(int argc, char* argv[])
5       {
6           printf ("hello world\n");
7
8           return 0;
9       }
```

```
(gdb) run
Starting program: /home/ceng328/code1
hello world

Program exited normally.
(gdb) break 6
Breakpoint 1 at 0x400503: file code1.c, line 6.
(gdb) run
Starting program: /home/ceng328/code1

Breakpoint 1, main (argc=1, argv=0x7fffffffe2c8)
at code1.c:6
6           printf ("hello world\n");
(gdb) step
hello world
8           return 0;
(gdb) continue
Continuing.

Program exited normally.
(gdb) quit
$
```

1. C Review II

- Exercise;
  - Run and debug code11.c without "datafile" file.
  - Analyze how **fp** behaves before and after **fopen** function.
  - Download datafile and analyze how **fp** behaves before and after **fopen** function again.

  - Run and debug the code from the last slide of 2nd Laboratory Experiment.
  - Modify **command** variable to alter the execution of the program.