Programming in C and C++ Past Paper

Basics

- y2021p4q1 (a)
  - static, extern
- y2016p3q1 (a)
  - char and string literal
- y2015p3q1 (a)
  - inline and drawback
- y2022p4q4 (d)
  - cast, endian

bit-level

- y2021p4q1 (b)
- y2015p3q1 (b)
  - function `receive_bit`, unsigned

Memory Organisation

- y2007p3q4 (a)
- y2021p4q1 (d)
  - storage and efficiency, interpreter
- y2022p4q4 (b)
  - string, caller vs callee

Data Structures

- y2015p3q2 (a)
  - C pointers vs C++ references [syntax, initialisation, mutation and safety]
- y2020p4q2 (a)
  - pointers and arrays
- y2022p4q5 (a,b)
  - FIFO, singly-linked list, `union`
- y2021p4q1 (c)
  - linked list, continue, bugs finding
- y2010p3q6
  - XOR linked list
- y1995p5q5
  - algorithms, bugs finding
- y2017p23q1 (a)

    - string, bugs finding
- y2022p4q4
    - string

Behaviour and Semantics

Implementation-defined (one), unspecified (a set of possibilities), undefined behaviour

- y2015p3q2 (c)
    - defined vs unspecified
- y2016p3q2 (a)
    - unspecified behaviour and its advantage
- y2020p4q2 (b)
    - advantage and disadvantage of implementation-defined operations
- y2019p4q2 (a)
    - string, `strlen`
- y2017p23q1 (a)
    - signed integer overflow `INT_MAX+1`
- y2016p3q1 (d)
    - arithmetic, signed integer underflow `-INT_MAX`
- y2015p3q1 (d)
    - buffer overflow, stack var out of scope, deref NULL pointer (from `malloc` heap)
    - access to uninitialized vars (stack/heap), etc

Cache-aware

- y2019p4q2 (b)

Object and Class

- y2007p3q4 (b,c)
    - C `struct` and C++ class
- y2017p23q2
- y2020p4q2 (d,e)
    - C++ virtual, RAII
- y2019p4q2 (c)
- y2016p3q2 (b,c)i
- y2022p4q5 (c)

Linking

- y2022p4q4 (c)
    - header / source file
- y2015p3q2 (b)

    - C and C++ linking

Exception and Template

- y2020p4q2 (c)
    - C++ Template vs Java Generics
- y2016p3q2 (c)iii
- y2015p3q1 (c)
- y2022p4q5 (d)

Debugging

- y2016p3q2 (c)ii
- y2016p3q1 (b,c)
    - Functions and Preprocessor
- y2015p3q2 (d)
    - debugger `lldb`, breakpoints and watch-points, symbol tables