UNIVERSITY OF CAMBRIDGE

COMPUTER SCIENCE TRIPOS Part IB

Monday 1 June 2015 1.30 to 4.30 pm

COMPUTER SCIENCE Paper 3

*Answer **five** questions.*

*Submit the answers in five **separate** bundles, each with its own cover sheet. On each cover sheet, write the numbers of **all** attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags
Rough work pad

SPECIAL REQUIREMENTS
Approved calculator permitted

1 Programming in C and C++

A spacecraft arrives at Mars, but its memory has been corrupted by radiation en route. Luckily, it can receive updates one bit at a time using a predefined C function `short receive_bit(void)`, that when called will return either 1 or 0. The stream of bits for a value is transmitted in unsigned big-endian byte order: for example, a 16-bit value of `125` would be `0000000001111101`. Assume the `int` type is 32 bits.

(*a*) Explain the meaning of the `inline` keyword on C function declarations, and a potential drawback of using it. [2 marks]

(*b*) Using `receive_bit()`, define a function `receive_int()` that decodes and returns a 32-bit value from the sequence of received bits. [4 marks]

(*c*) Build a more general decoding function `receive` using a C++ template with two parameters that specify the number of bits to decode and a datatype for the decoded value. Use this to write two template instantiations that decode an 8-bit value into a `short` and a 32-bit value into an `unsigned long`. [6 marks]

(*d*) Find and explain *four* instances of undefined behaviour that could result from compiling and running the C code below with different command-line arguments. The `strcpy(dst,src)` function copies a zero-terminated C string from the `src` buffer to the `dst` buffer. The `putchar(c)` function outputs a character `c` to the console. You can assume that the standard C header prototypes have been included for `<stdio.h>`, `<stdlib.h>` and `<string.h>`. [8 marks]

```
1.      char *show_instruction(int msg) {
2.        char buf[6];
3.        int fuel;
4.        if (msg == 1 && fuel--) {
5.          strcpy(buf, "THRUST");
6.          return buf;
7.        } else if (msg == 2) {
8.          char *msg = (char *)malloc(100);
9.          strcpy(msg, "DEPLOY_PARACHUTE");
10.         return msg;
11.       }
12.     }
13.
14.     int main(int argc, char **argv) {
15.        char *msg;
16.        msg = show_instruction(argc);
17.        putchar(msg[0]);
18.        return 0;
19.     }
```

2 Programming in C and C++

In this question, you may use short fragments of C or C++ code to complement your answer, where appropriate. Give a brief explanation of the following aspects of C and C++:

(*a*) The differences between C pointers and C++ references. [*Hint:* Consider issues of syntax, initialisation, mutation and safety in your answer.] [5 marks]

(*b*) How C and C++ object files may be safely linked with each other, and the limitations in doing so for some C++ features. [5 marks]

(*c*) The difference between implementation-defined behaviour and unspecified behaviour in the C standard, and an example of each sort of behaviour. Also briefly discuss why these loosely specified behaviours exist in the C standard instead of being strictly defined. [5 marks]

(*d*) The role of a debugger such as LLVM's `lldb` in locating bugs in C code, including the use of breakpoints and watchpoints in an interactive debugger and how symbol tables are useful. [5 marks]

3 Compiler Construction

Consider the following context-free grammar of expressions

$$E ::= n \mid (E,\ E)$$

where n ranges over integers.

(*a*) Present a right-most derivation of the expression ((21, 18), 17). [2 marks]

(*b*) List the LR(0) items for this grammar. [2 marks]

(*c*) Describe the states of the deterministic finite automata associated with an LR(0) parser for the grammar presented above. Explain your method of constructing these states. [4 marks]

(*d*) Describe the calculation of the *goto* function associated with an LR(0) parser for the grammar above. How is the goto function used by the parser? [4 marks]

(*e*) Carefully describe the LR(0) parsing associated with your derivation in (*a*). That is, show each transition of the parser and how it performs *shift* and *reduce* operations. [8 marks]

4 Compiler Construction

(*a*) Consider a programming language with nested function declarations that allows only *first-order* functions. That is, functions are not treated as values and can neither be passed as arguments nor returned by functions.

Lambda lifting and *static links* are two common methods of implementing such a language using a run-time stack. Describe these methods and discuss their advantages and disadvantages. [6 marks]

(*b*) Now suppose we are dealing with a programming language that supports higher-order functions that can be passed as arguments and returned as results. Give an example, in ML-like pseudo-code, where the techniques that you have described in (*a*) can no longer be used. Justify your answer. [6 marks]

(*c*) Carefully explain the techniques you might use to compile the example that you presented in (*b*). [8 marks]

5 Concepts in Programming Languages

(*a*) Identify three programming or implementation concepts stemming from Algol or Fortran which are still mainstream ideas, noting whether they arose from Algol or Fortran. [3 marks]

(*b*) Give two comparable now-mainstream ideas, but excluding ideas directly involving typing, which arose other than from Algol and Fortran. [2 marks]

(*c*) At machine-code level, a structured value typically contains some form of tag indicating which of several variant forms the data may take (e.g. 'leaf' versus 'branch'); this determines how subsequent fields are to be interpreted (e.g. as pointers, strings or floating-point values).

(*i*) Give code to show how type declarations in Java, ML and Pascal may be used to describe such structured values, following the above description as closely as possible. [5 marks]

(*ii*) For one of your type declarations identify a source of type unsafety that can arise. [2 marks]

(*d*) What is the principal difference between statically typed and dynamically typed languages? Identify one early and one more modern statically typed language. Identify one early and one more modern dynamically typed language. [3 marks]

(*e*) What is the principal difference between type-safe and type-unsafe languages? Identify one type-safe language and one type-unsafe language, from any era. [2 marks]

(*f*) A downcast is a cast from a reference of a base class to a reference of one of its derived classes. Discuss the extent to which Java and C++ are statically typed and type-safe given that a downcast may fail at run-time. [3 marks]

6 Further Java

Consider the following program extract from a server application:

```
ServerSocket ss = new ServerSocket(2311);
while(true) {
 Socket s = ss.accept();
 ObjectOutputStream oos =
      new ObjectOutputStream(s.getOutputStream());
 oos.writeObject(new Date());
 s.close();
}
```

(*a*) Describe the difference between `Socket` and `ServerSocket`, including the operation of the `accept()` method. [3 marks]

(*b*) Write a client program which connects to the server, receives an object, and prints the result of the object's `toString()` method. The three checked exceptions should be handled individually and an appropriate message printed. Ignore unchecked exceptions such as `OutOfMemoryError`. [5 marks]

(*c*) Does the execution of the `toString()` method in Part (*b*) pose a security risk to the client? Explain your reasoning. [2 marks]

(*d*) Rewrite the server to support multiple simultaneous client connections. Every second, the server should send a new `Date` object followed by an `Integer` object to all clients. The `Integer` object should contain the number of connected clients. [10 marks]

7 Prolog

This question explores how we might use Prolog to match Regular Expressions.

We represent the sequence to be matched in Prolog using a list of atoms. For example, `aaba` would be represented as the list `[a,a,b,a,end]` using the atom `end` to encode the end of the string explicitly.

A simple scheme for writing Regular Expressions uses a single character as itself and uses the plus symbol (+) to indicate that there should be one or more instances of the previous character. In this question we consider the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$ which means one or more occurrences of `a`, followed by one or more occurrences of `b`, followed by a single occurrence of `a`.

(*a*) Draw a state machine which is capable of matching the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$. Clearly indicate the start and finish states. [2 marks]

(*b*) Define a predicate `t(A,B,C)` which encodes the transitions of your state machine. `t(A,B,C)` should be true if there is a transition from state `A` to state `B` when we see a character `C`. Indicate which of your definitions are facts and which are rules. [2 marks]

(*c*) Predicates for *testing* a solution do not always work when *generating* solutions. Demonstrate this by writing a Prolog predicate `matches(L)` which *tests* if `L` represents a string which matches the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$. [5 marks]

(*d*) Why is your predicate `matches(L)` not a good solution for *generating* strings matching the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$? Describe a specific execution path in which a problem can occur. [3 marks]

(*e*) Describe a better strategy for *generating* strings matching the Regular Expression $\texttt{a}^+\texttt{b}^+\texttt{a}$ and provide an implementation. Clearly explain the approach you are using and why it is a sensible choice. [8 marks]

8 Software Engineering

You are advising the new Secretary of State for Work and Pensions following the 2015 election and the failure of the previous government's Universal Credit scheme for modernising welfare payments. The government's policy is to move away from a "big project" approach to system change and instead have a platform on which benefits can be rolled out, or old benefits modified, as needed. The new Secretary of State wants to develop a concept of how such a system will operate that will find favour with Cabinet colleagues.

Suppose as an example a minister decides to award a pension supplement to disabled people over 80 and their carers.

(*a*) Describe the processes and issues likely to be involved in:

(*i*) establishing a business requirement and likely cost; [4 marks]

(*ii*) determining what changes are needed to platform applications or other systems; [4 marks]

(*iii*) deciding whether such changes should be prototyped in the hope of avoiding unforeseen consequences or for other reasons; [4 marks]

(*iv*) assessing how to maintain any important emergent properties of the system as a whole (such as security, safety and fairness). [4 marks]

(*b*) What lessons might be drawn from the failures of other public-sector systems in the past? [4 marks]

END OF PAPER