1 Concepts in Programming Languages (am21)

(a) static void fastFilter(ArrayList<Datum> a, Predicate<String> p) {
    int j = 0;
    for (int i = 0; i < a.size(); i++) {
        if (p.test(a.get(i).key)) a.set(j++, a.get(i));
    }
    a.removeRange(j, a.size());
}

The above method fastFilter operating on ArrayList<Datum> was written by a Java programmer, perhaps misguidedly in the interests of speed. It uses method removeRange to remove the final a.size()-j elements from a. We wish to enable fastFilter also to operate on ArrayList<DatumDash> where DatumDash inherits from Datum.

(i) Give code for a modified version using subtype polymorphism, justifying any changes and highlighting any unresolvable difficulties. [5 marks]

(ii) Similarly, what changes, if any, would be required if the original fastFilter instead used Java classic [] arrays? Again justify your answer. [3 marks]

(iii) Give a definition of fastFilterInner which uses generic polymorphism, along with a subtype-polymorphic wrapper whose signature matches that used in Part (a)(i). [Hint: consider moving the use of .key.] [4 marks]

(b) class Myst { // A Mystery?
    private Supplier<Int> act; // Java's name for void->Int
    private Myst(Supplier<Int> a) { act = a; }
    static Myst R(Int x) { return new Myst(()->x); }
    static Myst B(Function<Int,Myst> f) { return f.apply(act.get()); }
    static Myst primIn = new Myst( () -> SystemIO.readInt() );
    static Myst primOut(Int x) {
        return new Myst(() -> { SystemIO.println(x); return x;});
    }
    static void exec(Myst x) { x.act.get(); }
}

The class Myst is coded in a language resembling Java. What concept does it model and what does the code below conceptually do if executed in its scope?

Myst one = primIn.B(x -> primOut(x+1));
Myst two = R(2).B(y -> primIn.B(x -> primOut(x+y)));
Myst three = one.B(x -> two).B(x -> one);
exec(three);