**SE 504 (Formal Methods and Models)**
**Spring 2020**
**HW #1: Developing Predicates and Specifications**
**Due: 7:20pm, Thursday, Feb. 6**

The problems on the next page ask you to develop boolean expressions that describe predicates, in some cases to be placed in Hoare triples intended to specify programs. As elements in such expressions, you are free to use any of the following in any of the usual ways:

1. literals (e.g., `5`, `false`)
2. variables (e.g., $x$, $p$)
3. the boolean operators $\equiv$ (equivales), $\not\equiv$ (not equivales), $\wedge$ (conjunction), $\vee$ (disjunction), $\Rightarrow$ (implication), $\Leftarrow$ (consequence), and $\neg$ (negation)
4. the arithmetic operators $+$ (addition), $-$ (subtraction), $\cdot$ (multiplication), / (or $\div$) (division), `min` (minimum), and `max` (maximum)
5. the relational operators $=$ (equals), $\neq$ (not equals), $<$ (less than), $\leq$ (less than or equal to), $>$ (greater than), and $\geq$ (greater than or equal to)
6. *quantification* over any operator that is both symmetric (i.e., commutative) and associative, including $+$, $\cdot$, `min`, `max`, $\wedge$, and $\vee$ (the last two of which give rise to the *universal* and *existential* quantifiers $\forall$ and $\exists$, respectively).
7. the "counting" quantifier, as in $(\#x \mid R : P)$, which is shorthand for $(+x \mid R \wedge P \ : \ 1)$.

To make your answers more intelligible, in some cases it may help to define one or more predicates (or other functions) and to refer to them elsewhere within the answer. As an example, consider the problem of specifiying a program that, assuming $n > 2$, establishes $m$'s value to be the largest prime number less than $n$. Here is a good answer:

$|[$ con $n : \mathbb{N}$;
var $m : \mathbb{N}$;
$\{ \ n > 2 \ \}$
$m \ := \ ?$
$\{ \ m < n \ \wedge \ \text{isPrime}.m \ \wedge \ (\forall i \mid m < i < n : \neg \text{isPrime}.i) \ \}$
$]|$

where $\text{isPrime}.r \ ::= \ r > 1 \ \wedge \ \neg(\exists k \mid 1 < k < r \ : \ k|r)$,
where $a|b \ ::= \ (\exists c : \mathbb{N} \mid : \ a \cdot c = b)$.

Here we refer to the predicate *isPrime* twice in the Hoare Triple, and define it thereafter. The definition of *isPrime* itself refers to the "is a divisor of" predicate (denoted by a vertical bar), the definition of which is given thereafter.

The assumption here is that all variables (including dummies) are of type $\mathbb{N}$ (or `int`, if you prefer). Note that the symbol ::= is intended to be read as "is defined as".

Several of the problems involve arrays. The length of an array $b$ is denoted by $\#b$ (or by $b.length$, if you prefer); hence, the index range of $b$ is $[0..\#b)$.

For each of the following (informally stated) predicates, express it formally, using a boolean expression.

**1.** There is at least one prime number in the interval $[k..m]$.

**2.** There are at least $n$ prime numbers in the array segment $b[k..m)$.
(*Hint:* Use the counting quantifier.)

**3.** If there is an occurrence of a prime number in array $b$, the first such occurrence is somewhere in the segment $b[k..m)$.

**4.** Every occurrence of zero in array $b$ is immediately followed by an occurrence of a positive odd number.

**5.** The number of distinct values occurring in array $b$ is $m$.
(*Hint:* Every value that occurs in $b$ has a first occurrence.)

**6.** The binary operator $\oplus : A \times A \to A$ is associative. (Exactly what set $A$ refers to is irrelevant.)

---

For each of the following (informally stated) program specifications, express it formally, in a form as in the example on the previous page.

**7.** Set (boolean variable) $q$'s value to the value of the proposition that the sum of the elements in array $b$ is negative.

**8.** Assuming that array $b$ contains only occurrences of 0s and 1s, interpret it as a binary numeral (with the least significant digit at location zero), setting $m$'s value to the number represented by $b$.

**9.** Let $a$ and $b$ be two arrays containing values of the same type. Set $k$ to the smallest value for which $a[k]$ does not occur in $b$.

**10.** Rearrange the elements of array $b$ so that all, and only, the elements in $b[0..k)$ are less than $x$.
*Hint 1:* Use a rigid variable referring to $b$'s initial value.
*Hint 2:* Define (and use) a predicate that indicates whether one array is a permutation (i.e., rearrangement) of another. The counting quantifier could help here.