BU CAS CS 320: Concepts of Programming Languages

Midterm-2 Examination

Instructor: Hongwei Xi

June 23, 2023

Name:__________________________ Score:__________________________

No.	Points	Answer	Score
01.	10		
02.	10		
03.	20		
04.	20		
05.	20		
06.	20		
Total	100		

No computer is allowed!

(And your phone is considered a computer.)

Question 1 `(*`

```
//
// HX-2023-06-23: 10 points
// 10 points for stream_take
//
Given a stream fxs, stream_take(fxs, n)
returns another stream containing the first
n items in fxs (or all the elements of fxs if
fxs contains fewer than n elements).
//
fun
stream_take
(fxs: 'a stream, n: int): 'a stream = ...
//
*)
```

Question 2 (*

```
//
// HX-2023-06-23: 10 points
// 10 points for stream_drop
//
Given a stream fxs, stream_drop(fxs, n)
returns another stream containing all but the
first n elements of fxs. Note that the returned
stream is empty if fxs contains fewer than n
elements.
//
fun
stream_drop
(fxs: 'a stream, n: int): 'a stream = ...
//
*)
```

Question 3 (*

```
//
// HX-2023-06-23: 20 points
//
Given a stream fxs of real numbers a0, a1, a2, ...
and a real number x0, stream_evaluate(fxs, x0)
returns another stream of real number that enumerates
all of the following partial sums:
a0, a0 + a1*x0, a0 + a1*x0 + a2*x0^2, ...
The general form of the enumerated sums is given as follows:
(a0 + a1*x0 + a2*x0^2 + ... + an * x0^n)
//
Assume:
a0 = 0, a1 = 1, a2 = -1/2, a3 = 1/3, a4 = -1/4, ...
Then we have ln2 = stream_evaluate(fxs, 1.0) // see Assign03
//
fun
stream_evaluate
(fxs: real stream, x0: real): real stream = ...
//
*)
```

Question 4 (*

```
//
HX-2023-06-23: 20 points
//
A non-empty sequence of numbers forms a
"drawdown" if every number in the sequence does not
exceed the first one. A maximal drawdown is one that
is not contained in any longer drawdowns.
Please implement a function stream_drawdowns that takes
an infinite stream fxs of integers and returns a stream
enumerating all the maximal drawdowns in fxs.
//
fun
stream_drawdowns(fxs: int stream): int list stream = ...
//
*)
```

Question 5 (*

```
//
// HX-2023-06-23: 20 points
//
A sequence xs of integers captures '231'
if there are three integers a, b, and c
appearing as a subsequence of xs satisfying
c < a < b. NOTE that a, b, and c do not have
to appear consecutively in xs.
//
For instance, [1,3,4,2] does capture '231'
For instance, [1,2,4,3] does not capture '231'
For instance, [1,2,3,4] does not capture '231'
//
fun
perm_capture_231(xs: int list): bool = ...
//
*)
```

Question 6 (*

```
//
// HX-2023-06-23: 20 points
//
Given a list xs and a natural number k0,
perm_counting_out(xs, k0) returns a permutation
of xs where the elements are listed according to
the order they are "counted out" in the following
process of counting:
//
Counting of the elements xs goes left to right
and the first count is 0. When the count reaches
k0, the element being counted is removed (that is,
the element is counted out) and counting starts again
with the following element. If counting reached the
last element remaining in the list, then the next element
to be counted is the first element in the list. Counting
stops when all the elements are counted out.
//
For instance,
perm_counting_out([1,2,3], 0) = [1,2,3]
perm_counting_out([1,2,3], 1) = [2,1,3]
perm_counting_out([1,2,3], 2) = [3,1,2]
perm_counting_out([1,2,3], 3) = [1,3,2]
perm_counting_out([1,2,3,4], 1) = [2,4,3,1]
perm_counting_out([1,2,3,4], 3) = [4,1,3,2]
//
fun
perm_counting_out(xs: int list, k0: int): int list = ...
//
*)
```