Niezawodne Systemy Informatyczne

Lista 1

PRZEMYSŁAW KOBYLAŃSKI

Rozwiąż samodzielnie poniższe zadania i przedstaw prowadzącemu najpóźniej na 5. zajęciach.

Przy każdym z zadań podano maksymalną do uzyskania liczbę punktów.

Za zadanie otrzymuje się:
- 100% punktów gdy program **gnatprove** udowodnił wszystkie asercje (*Assert*), kontrakty (*Post*), kontrole indeksów (*index check*) i zakresów (*overflow check*) oraz nie użyto założeń (*Assume*);
- 75% punktów gdy program **gnatprove** udowodnił wszystkie asercje, kontrakty, kontrole indeksów;
- 50% punktów gdy program **gnatprove** udowodnił wszystkie kontrakty i kontrole indeksów (nieudowodniona asercja *Assert* będzie uznawana za użycie założenia *Assume* i wymaga uzasadnienia słownego podczas zaliczania).

Każde użyte założenie *Assume* należy udowodnić prowadzącemu podczas zaliczania listy ale zasadność jego użycia może być zakwestionowana przez prowadzącego a ma on głos rozstrzygający.

Wskazówka: poczytaj o parametrach polecenia **gnatprove** (*level*, *steps*, *timeout* i innych).

Zadanie 1 (16 pkt)

Poniżej zamieszczono fragment procedury **Smallest_Factor**, która znajduje najmniejszy dzielnik **Factor** liczby całkowitej **N** większej od 1 i dzieli **N** przez ten dzielnik.

```
procedure Smallest_Factor (N : in out Positive; Factor : out Positive)
  with
  SPARK_Mode,
  Pre => N > 1,
  Post => (Factor > 1) and
          (N'Old / Factor = N) and
          (N'Old rem Factor = 0) and
          (for all J in 2 .. Factor - 1 => N'Old rem J /= 0)
is
begin
   …
end Smallest_Factor;
```

Napisz procedurę **Smallest_Factor** i udowodnij ją programem **gnatprove**.

Napisz procedurę **Main**, która testuje poprawność działania procedury **Smallest_Factor**.

Zadanie 2 (8 pkt)

Poniżej przedstawiono specyfikację pakietu **Poly**, który zawiera funkcję **Horner** wyliczającą zgodnie ze schematem Hornera wartość wielomianu o współczynnikach całkowitych dla argumentu całkowitego.

package Poly **with** SPARK_Mode **Is**

  type Vector **is array** (Natural **range** <>) **of** Integer;

  function Horner (X : Integer; A : Vector) **return** Integer;

end Poly;

Przyjmij dla uproszczenia, że liczba **A(A'First)** jest wyrazem wolnym, **A(A'First + 1)** jest współczynnikiem przy **X**, **A(A'First + 2)** jest współczynnikiem przy **X ** 2**, itd.

Napisz treść pakietu **Poly** zamieszczając w nim asercje i niezmienniki pętli tak aby program **gnatprove** udowodnił, że funkcja **Horner** poprawnie liczy wartość wielomianu zgodnie ze schematem Hornera.

Napisz procedurę **Main**, która testuje poprawność działania funkcji **Horner**.

Zadanie 3 (4 pkt)

Poniżej przedstawiono specyfikację pakietu **Selection**, który zawiera procedurę **Sort** sortującą daną tablicę algorytmem SELECTION SORT.

```
package Selection with SPARK_Mode is

   type Arr is array (Integer range <>) of Integer;

   function Sorted (A : Arr) return Boolean
   is (for all I in A'First .. A'Last - 1 => A(I) <= A(I + 1))
   with
     Ghost,
     Pre => A'Last > Integer'First;

   procedure Sort (A : in out Arr)
   with
     Pre => A'First in Integer'First+1..Integer'Last-1 and
            A'Last in Integer'First+1..Integer'Last-1,
     Post => Sorted (A);

end Selection;
```

Napisz treść pakietu **Selection** tak aby program **gnatprove** udowodnił wszystkie kontrakty, asercje, niezmienniki pętli i inne użyte w specyfikacji i treści.

Napisz procedurę **Main**, która testuje poprawność działania procedury **Sort**.