Dowodzenie poprawności programów

Wstęp do Informatyki i Programowania

Maciek Gębala

21 listopada 2024

Notatki

Mnożenie

Chcemy pomnożyć dwie liczby całkowite a i b posługując się tylko dodawaniem, dzieleniem przez 2 i sprawdzaniem nieparzystości.

$x \leftarrow a$, $y \leftarrow b$, $z \leftarrow 0$ $\{x = a \wedge y = b \wedge z = 0\}$
while $x \neq 0$ **do**
  $\{z + xy = ab\}$
  if $odd(x)$ **then**
    $z \leftarrow z + y$
  end if
  $\{(odd(x) \wedge z - y + xy = ab) \vee (\neg odd(x) \wedge z + xy = ab)\}$
  $x \leftarrow \lfloor x/2 \rfloor$
  $\{z + 2xy = ab\}$
  $y \leftarrow y + y$
  $\{z + xy = ab\}$
end while
$\{z = ab\}$

Notatki

Mnożenie

Niezmiennik $z + xy = ab$ inicjalizuje się dobrze po początkowych przypisaniach $x \leftarrow a$, $y \leftarrow b$ i $z \leftarrow 0$.

Z niezmiennika $z + xy = ab$ i zaprzeczenia warunku pętli $x = 0$ wynika natychmiast warunek końcowy $z = ab$.

Niezmiennik $z + xy = ab$ odtwarza się po każdym obrocie pętli. Jeśli na początku pętli x jest parzyste, to pierwsza instrukcja jest równoważna pustej i w oczywisty sposób niezmiennik pozostaje prawdziwy. Jeśli natomiast na początku pętli x jest nieparzyste, to najpierw wykona się instrukcja $z \leftarrow z + y$, po której będzie spełniony warunek $z - y + xy = ab$. Teraz wykonanie instrukcji $x \leftarrow \lfloor x/2 \rfloor$ spowoduje, że będzie spełniony warunek $z - y + (2x + 1)y = ab$, co jest równoważne warunkowi $z + 2xy = ab$, co po przypisaniu $y \leftarrow y + y$ doprowadza nas z powrotem do niezmiennika $z + xy = ab$, a to jest odtworzony niezmiennik.

Notatki

Mnożenie

$x \leftarrow a$, $y \leftarrow b$, $z \leftarrow 0$
while $x \neq 0$ **do**
  if $odd(x)$ **then**
    $z \leftarrow z + y$
  end if
  $x \leftarrow \lfloor x/2 \rfloor$
  $y \leftarrow y + y$
end while

Jest oczywiste, że jeśli $a \geqslant 0$ to pętla się zakończy.

Co się jednak stanie gdy $a < 0$?

Prześledźmy na tablicy działanie programu dla $a = -5$ i $b = 2$.

Program się zapętla, gdyż $\lfloor -1/2 \rfloor = -1$. Zatem program jest tylko częściowo poprawny (niezmiennik pętli $z + xy = ab$ zawsze jest prawdziwy).

Notatki

Wyszukiwanie binarne

Załóżmy, że w tablicy $A(1:n)$ mamy posortowane niemalejąco liczby całkowite i chcemy sprawdzić, czy jest w niej element x.

$l \leftarrow 1,\ p \leftarrow n$
while $l < p$ **do**
  $s \leftarrow \lfloor (l+p)/2 \rfloor$
  if $x > A[s]$ **then**
    $l \leftarrow s + 1$
  else
    $p \leftarrow s$
  end if
end while
$jest \leftarrow x = A[l]$

Nie sprawdzamy nigdzie, czy przypadkiem $A[s] = x$, tylko doprowadzamy za każdym razem do momentu, w którym badany przedział jest jednoelementowy.

Notatki

Wyszukiwanie binarne

Ustalmy, ze predykat $Sort(A)$ jest prawdziwy jeśli tablica A jest posortowana niemalejąco.

Dopiszmy teraz do pętli niezmiennik

$l \leftarrow 1,\ p \leftarrow n$
while $l < p$ **do**
  $\{Sort(A) \wedge (1 \leqslant l \leqslant p \leqslant n) \wedge (\exists_{1 \leqslant k \leqslant n}\, A[k] = x \iff \exists_{l \leqslant k \leqslant p}\, A[k] = x)\}$
  $s \leftarrow \lfloor (l+p)/2 \rfloor$
  if $x > A[s]$ **then**
    $l \leftarrow s + 1$
  else
    $p \leftarrow s$
  end if
end while
$jest \leftarrow x = A[l]$

Notatki

Wyszukiwanie binarne

Niezmiennik

$$Sort(A) \wedge (1 \leqslant l \leqslant p \leqslant n) \wedge (\exists_{1 \leqslant k \leqslant n}\, A[k] = x \iff \exists_{l \leqslant k \leqslant p}\, A[k] = x)$$

mówi nam, że

1. tablica jest posortowana cały czas,
2. indeksy l oraz p są w zakresie indeksów tablicy i l nie przekracza p
3. jeśli element x jest gdzieś w tablicy, to jest między indeksami l oraz p.

Zauważmy, że niezmiennik inicjalizuje się dobrze, jeśli tylko $1 \leqslant n$ (czyli tablica jest niepusta).

Notatki

Wyszukiwanie binarne

Ze względu na to, że dla $k > 1$ zachodzi $1 \leqslant \lfloor k/2 \rfloor < k$otrzymujemy nierówność $l \leqslant s < p$ dopóki $1 \leqslant l < p$ i ta ostra nierówność jest bardzo istotna. Zauważmy więc, że przypisując zmiennej l wartość $s+1$ otrzymujemy wartość ostro większą od l, a przypisując zmiennej p wartość s otrzymujemy wartość mniejszą ostro od p. Zatem $p - l$ ma wartości w zbiorze liczb naturalnych i maleje z każdym obrotem pętli. Czyli pętla zawsze się zakończy.

Jeśli $x > A[s]$, to ze względu na przechodniość relacji większości mamy $x > A[k]$ dla $1 \leqslant k < s$, więc jeśli istnieje x w tablicy, to istnieje między $s+1$ a p.

Jeśli zaś $x \leqslant A[s]$, wówczas jeśli istnieje x w tablicy, to w szczególności jest w przedziale od 1 do s, zatem przypisanie $p \leftarrow s$ nadal utrzymuje w prawdziwości ten warunek.

Notatki

Wyszukiwanie binarne

Pozostaje pytanie czy zaprzeczenie warunku pętli w połączeniu z niezmiennikiem da nam końcowy warunek

$$A[l] = x \iff \exists_{1 \leqslant k \leqslant n} A[k] = x?$$

Zaprzeczenie warunku to $l \geqslant p$. W koniunkcji z niezmiennikiem daje nam $l = p$. Z niezmiennika wiemy, że warunek $\exists_{1 \leqslant k \leqslant n} A[k] = x$ jest równoważny warunkowi $\exists_{l \leqslant k \leqslant p} A[k] = x$, ale że $l = p$, to jest to równoważne warunkowi $A[l] = x$.

Tak znaleziona wartość x będzie zawsze pierwsza od lewej. Dlaczego?

Notatki

Wyszukiwanie binarne

Załóżmy, że instrukcję

if $x > A[s]$ **then**
  $l \leftarrow s + 1$
else
  $p \leftarrow s$
end if

zastąpimy instrukcją

if $x \geqslant A[s]$ **then**
  $l \leftarrow s$
else
  $p \leftarrow s - 1$
end if

Sytuacja wygląda na zupełnie symetryczną.

Jest to bardzo częsty błąd popełniany przy kodowaniu tego algorytmu. Program może się zapętlić. Kiedy i dlaczego?

Notatki

Notatki

Notatki