Pracovní ukázka vstupního testu DSA 1.

Celkem můžete získat 6 bodů,
k úspěšnému vyřešení testu je nutno získat alespoň 4 body.

V úloze 1. získáte 1 bod za každou správně určenou hodnotu.
V úlohách 2. a 3. získáte 1 bod za každý správně doplněný řádek kódu.

1. Rekurze

Napište, jakou hodnotu vrátí volání funkce `rek(10,4)` a `rek(3,7)`.

```
int rek(int x, int y) {
  if (x < y) return rek(x,y-1)+1;
  return y;
}
```

rek(10, 4) =	rek(3, 7) =

2. Testování dat

Je dán neprázdný jednosměrný spojový seznam celých čísel, na jehož první prvek ukazuje ukazatel (reference) **hlava**. Prvek seznamu je typu **Prvek** a obsahuje celočíselnou složku **hodnota** a ukazatel **next** na další prvek. Dále je dáno jednorozměrné pole **pole** celých čísel.
Následující kód představuje tělo funkce, která testuje, zda je obsah pole i seznamu identický (t.j. seznam i pole obsahuje tytéž hodnoty ve stejném pořadí).
Doplňte chybějící části kódu, každý řádek obsahuje jen jeden příkaz nebo jen jednu hlavičku cyklu/podmínky.

```
 Prvek ukPrvek = hlava;
                                                    (doplň)
    if ((ukPrvek == null) || (ukPrvek.hodnota != pole[i]))
       return false;
                                                    (doplň)
  }
 return (ukPrvek == null) ;
} // konec tela funkce
```

3. Přesun dat

Jsou dány dva obousměrné neprázdné spojové seznamy celých čísel. Prvek každého seznamu je typu **Prvek**, obsahuje celočíselnou složku **hodnota**, ukazatel **next** na na další prvek a ukazatel **prev** na předchozí prvek. Na poslední prvek prvního seznamu ukazuje ukazatel (reference) **konec1**, na první prvek druhého seznamu ukazuje ukazatel **hlava2** (žádný ukazatel ukazující na konec druhého seznamu není definován).
Následující kód připojí na konec prvního seznamu celý druhý seznam a nastaví ukazatel **konec1** na poslední prvek prodlouženého prvního seznamu a ukazatel **hlava2** nastaví na hodnotu **null**.
Doplňte chybějící části kódu, každý řádek obsahuje jen jeden příkaz nebo jen jednu hlavičku cyklu/podmínky.

```
                                                    (doplň)
  hlava2.prev = konec1;
                                                    (doplň)
      konec1 = konec1.next;
  hlava2 = null;
```

Řešení

1. Rekurze

rek(10,4) = **4**,
rek(3, 7) = rek(3, 6) + 1 = rek(3, 5) + 1 +1 = rek(3, 4) + 1 + 1 + 1 = rek(3, 3) + 1 + 1 + 1 + 1 = ¨
= 3 + 1 + 1 + 1 + 1 = **7**.

2. Testování dat

```
  Prvek ukPrvek = hlava;
  for (int i = 0; i<pole.length; i++) {
     if ((ukPrvek == null) || (ukPrvek.hodnota != pole[i]))
        return false;
    ukPrvek = ukPrvek.next;
    }
  return (ukPrvek == null) ;
 } // konec tela funkce
```

3. Přesun dat

```
  konec1.next = hlava2;
  hlava2.prev = konec1;
  while (konec1.next != null)
      konec1 = konec1.next;
  hlava2 = null;
```

Pracovní ukázka vstupního testu DSA 2.

Celkem můžete získat 6 bodů,
k úspěšnému vyřešení testu je nutno získat alespoň 4 body.

V úloze 1. získáte 1 bod za každou správně určenou hodnotu.
V úlohách 2. a 3. získáte 1 bod za každý správně doplněný řádek kódu.

1. Rekurze

Napište, jakou hodnotu vrátí volání funkce `rek(10,4)` a `rek(4,1)`.

```
int rek(int x, int y) {
 if (x < y) return 0;
 return rek(x-y,y)+1;
}
```

rek(10, 4) =	rek(4, 1) =

2. Testování dat

Je dán neprázdný jednosměrný spojový seznam celých čísel, na jehož první prvek ukazuje ukazatel (reference) **hlava**. Prvek seznamu je typu **Prvek** a obsahuje celočíselnou složku **hodnota** a ukazatel **next** na na další prvek. Dále je dáno jednorozměrne pole **pole** celých čísel.
Následující kód představuje tělo funkce, která testuje, zda každý prvek seznamu je menší než nejmenší prvek pole. Doplňte chybějící části kódu, každý řádek obsahuje jen jeden příkaz nebo jen jednu hlavičku cyklu/podmínky.

```
  int min = pole[0];
  for (int i = 1; i < pole.length; i++)
                                                    (doplň)
         min = pole[i];
  Prvek ukPrvek = hlava;
                                                    (doplň)
      if (ukPrvek.hodnota) >= min)
         return false;
      ukPrvek = ukPrvek.next;
   }
  return true;
 } // konec tela funkce
```

3. Přesun dat

Je dán obousměrný neprázdný spojový seznam celých čísel, na jehož první prvek ukazuje ukazatel (reference) **hlava**. Prvek seznamu je typu **Prvek** a obsahuje celočíselnou složku **hodnota** a ukazatel **next** na na další prvek a ukazatel **prev** na předchozí prvek. Seznam je seřazen v neklesajícím pořadí, tj. každý jeho prvek je větší nebo roven svému předchůdci. Následující kód vymaže ze seznamu každý duplikát, tedy každý prvek, jehož hodnota je rovna hodnotě jeho předchůdce.
Doplňte chybějící části kódu, každý řádek obsahuje jen jeden příkaz nebo jen jednu hlavičku cyklu/podmínky.

Poznámka: Funkce **destroy** uvolní paměť alokovanou prvkem seznamu (v Javě by nebyla zapotřebí).

```
Prvek ukPrvek = head;
Prvek ukDalsi;
while (ukPrvek.next != null )
  if (ukPrvek.hodnota == ukPrvek.next.hodnota) {
                                                    (doplň)
    ukPrvek.next = ukDalsi.next;
    if (ukDalsi.next != null)
                                                    (doplň)
    destroy(ukDalsi);
  }
  else  ukPrvek = ukPrvek.next;
```

Řešení

1. Rekurze

rek(10,4) = rek(6, 4) + 1 = rek (2, 4) + 1 + 1 = 0 + 1 + 1 = **2**,

rek(4, 1) = rek(3, 1) + 1 = rek(2, 1) + 1 +1 = rek(1, 1) + 1 + 1 + 1 = rek(0, 1) + 1 + 1 + 1 + 1 =
= 0 + 1 + 1 + 1 + 1 = **4**.

2. Testování dat

```
  int min = pole[0];
  for (int i = 1; i < pole.length; i++)
      if (pole[i] < min)
          min = pole[i];
  Prvek ukPrvek = hlava;
  while (ukPrvek != null)
      if (ukPrvek.hodnota) >= min)
         return false;
      ukPrvek = ukPrvek.next;
   }
  return true;
 } // konec tela funkce
```

3. Přesun dat

```
  Prvek ukPrvek = head;
  Prvek ukDalsi;
  while (ukPrvek.next != null )
    if (ukPrvek.hodnota == ukPrvek.next.hodnota) {
      ukDalsi = ukPrvek.next;
      ukPrvek.next = ukDalsi.next;
      if (ukDalsi.next != null)
          ukDalsi.next.prev = ukPrvek;
      destroy(ukDalsi);
    }
   else  ukPrvek = ukPrvek.next;
```