Informatika – navazující magisterské studium
Přijímací zkouška z informatiky – 2017 – varianta A

Každá úloha je hodnocena maximálně 25 body.
Všechny své odpovědi zdůvodněte!

1. Čtverečkovou mřížku velikosti 2×10 políček chceme celou pokrýt deseti obdélníky o velikosti 1×2 políčka. Určete, kolika různými způsoby to můžeme udělat. Obdélníky můžeme libovolně natáčet.
Odvoďte obecný vztah pro počet různých pokrytí mřížky velikosti $2 \times N$ políček pomocí N obdélníků o velikosti 1×2 políčka.

2. Orientovaný graf je zadán počtem vrcholů a výčtem hran. Máte za úkol zjistit, zda tento graf obsahuje nějaký cyklus (tzn. cestu, která vede po hranách ve směru jejich orientace a vrací se do svého výchozího vrcholu). Popište algoritmus pro řešení tohoto úkolu a zdůvodněte jeho správnost. Odvoďte jeho asymptotickou časovou složitost.

3. Navrhněte deterministický konečný automat nad abecedou {0, 1}, který přijímá všechna slova liché délky, která obsahují právě dva znaky 1. Například slova 011, 10001, 0010100 automat přijme, zatímco slova 100, 1001, 10101 nepřijme. Přechodovou funkci automatu zapište ve tvaru tabulky a automat znázorněte ve tvaru přechodového diagramu. Navrhněte co nejjednodušší automat, tzn. takový, který bude mít minimální počet stavů.

4. Je dán následující program (obě zadání v Pascalu a v jazyce C jsou ekvivalentní):

```
program AA;
var a, b, k: integer;
begin
  b := 0;
  k := 3;
  while b <= 1000 do
    begin
    a := 0;
    while a <= b do a:=a+k;
    b := a;
    k := 8-k;
    end;
  writeln(b)
end.
```

```
main() /* AA */
{
  int a, b, k;
  b = 0;
  k = 3;
  while (b <= 1000)
  {
    a = 0;
    while (a <= b) a += k;
    b = a;
    k = 8-k;
  }
  printf("%d", b);
}
```

a) Určete, kolikrát se vykoná tělo vnějšího while-cyklu.
b) Určete výslednou hodnotu proměnné *b*.

Řešení přijímací zkoušky z informatiky – 2017 – varianta A

1. Počet různých způsobů pokrytí mřížky o velikosti 2 x *N* políček označíme *P(N)*. Hledáme hodnotu *P(10)*. V prvním sloupci mřížky umístíme obdélník 1x2 buď svisle, a pak za ním následuje *P(9)* možností pokrytí zbytku mřížky, nebo vodorovně (druhý nutně vodorovně pod něj) a za nimi následuje *P(8)* dalších možností. Celkově tedy *P(10) = P(9) + P(8)*, obecně *P(N) = P(N-1) + P(N-2)* pro všechna *N* > 2. Zřejmě platí *P(2) = 2, P(1) = 1*. Výsledný počet *P(N)* je tedy roven *N*-tému Fibonacciho číslu, výsledek *P(10)* snadno dopočítáme v deseti krocích: 1 2 3 5 8 13 21 34 55 **89**.

Jiný postup řešení pro *N*=10: Předpokládejme, že mřížka je umístěna vodorovně. Obdélníky 1x2 v ní mohou být umístěny buď svisle, nebo vodorovně – ale v tom případě jedině vždy dva nad sebou. Rozlišíme možné případy podle počtu dvojic vodorovných obdélníků:

- žádná vodorovná dvojice, 10 svislých	1 možnost
- jedna vodorovná dvojice, 8 svislých	9 možností
- dvě vodorovné dvojice, 6 svislých	(8 nad 2) = 28 možností
- tři vodorovné dvojice, 4 svislé	(7 nad 3) = 35 možností
- čtyři vodorovné dvojice, 2 svislé	(6 nad 2) = 15 možností
- pět vodorovných dvojic, žádný svislý	1 možnost

Celkem tedy dostáváme 1 + 9 + 28 + 35 + 15 + 1 = **89** možností.

2. Orientovaný graf je acyklický, právě když má topologické uspořádání. Pokusíme se tedy zadaný graf topologicky uspořádat z výsledku pak přímo plyne, zda graf obsahuje cyklus. V orientovaném acyklickém grafu vždy existuje vrchol, do něhož nevede žádná hrana (tvrzení snadno dokážeme sporem). Tomuto vrcholu dáme číslo 1 v topologickém uspořádání a odstraníme ho z grafu včetně všech hran, které z něj vedou. Tím dostaneme acyklický graf s počtem vrcholů o 1 menším. V něm postupujeme stejným způsobem dále a odebíraným vrcholům přidělujeme postupně další čísla. Pokud v některém kroku výpočtu nenajdeme vrchol, do kterého nevede žádná hrana, graf obsahuje cyklus. Pokud topologické třídění celého grafu úspěšně dokončíme, graf je acyklický. Asymptotická časová složitost závisí na tom, jaké použijeme datové struktury. Při vhodně volbě lze dosáhnout složitosti O(*N*+*M*), kde *N* je počet vrcholů a *M* je počet hran grafu. Existuje i jiný postup řešení založený na prohledávání grafu do hloubky.

3. Pomocí stavů automatu musíme počítat počet jedniček přečtených ze vstupu a evidovat paritu délky vstupního slova. Ve stavech A, B nebyla přečtena žádná 1, ve stavech C, D byla přečtena jedna 1, ve stavech E, F byly přečteny dvě 1, ve stavu G více než dvě 1 (přesný počet už není zajímavý). Minimalitu počtu stavů ověříme redukcí sestrojeného konečného automatu.

	0	1	
→ A	B	C	přečtena žádná 1, sudá délka
B	A	D	přečtena žádná 1, lichá délka
C	D	E	přečtena jedna 1, lichá délka
D	C	F	přečtena jedna 1, sudá délka
E	F	G	přečteny dvě 1, sudá délka
← F	E	G	přečteny dvě 1, lichá délka
G	G	G	přečteny více než dvě 1

4. V každém lichém průchodu vnějšího while-cyklu se hodnota proměnné *b* zvýší na nejbližší vyšší násobek tří, v každém sudém průchodu se zvýší na nejbližší vyšší násobek pěti. Začíná se zvyšovat od nuly, takže vždy po dvou průchodech se hodnota *b* zvýší o 5. Po 400 průchodech tak dosáhne hodnoty 1000, pro tuto hodnotu *b* se vnější while-cyklus ještě jednou vykoná a *b* při tom vzroste na nejbližší vyšší násobek tří, což je 1002.

a) **401**

b) **1002**