Teoretyczne podstawy informatyki
Zestaw zadań 7

dr Anna Ochab-Marcinek

1. Oszacować (w najbardziej niekorzystnum przypadku) czas działania fragmentu funkcji `SelectionSort` z poprzedniego zestawu:

```
small=i;
for(j=i+1 ; j<n ; j++)
        if(A[j]<A[small])
                small=j;
```

   Należy pozliczać razem ilość porównań i przypisań w tej pętli. Co jest naturalnym "rozmiarem" danych m, na których operuje fragment programu? Ile wynosi czas $T(m)$? [1]

2. Rozważyć program liczący silnię:

```
scanf("%d" , &n);
i=2;
fact=1;
while(i<=n)
        {
        fact=fact*i;
        i++;
        }
printf("%d␣\n" , fact );
```

   Niech dana na wejściu liczba n będzie jednocześnie rozmiarem danych wejściowych. Licząc po jednej jednostce czasu dla każdej instrukcji przypisania, instrukcji odczytu i zapisu oraz instrukcji warunkowej wykonywanej przed każdą iteracją pętli `while`, obliczyć czas działania programu [2].

3. Rozważyć funkcję `PowersOfTwo`:

```
int PowersOfTwo(int n)
        {
        int i;
        i=0;
        while(n%2 == 0)
                {
                n=n/2;
                i++;
```

```
                }
        return(i);
        }
```

Co robi funkcja? Jaki jest jej czas działania $O(n)$, gdy rozmiarem danych wejściowych jest wartość n? [3]

4. Rozważyć fragment programu:

```
scanf("%d", &n);
for(i=0;i<n;i++)
        for(j=0;j<n;j++)
                A[i][j]=0;
for(i=0;i<n;i++)
        A[i][i]=1;
```

Co program robi? Jaki jest jego czas działania $O(n)$? [4]

5. Jaki jest czas działania pętli `for` [5]: A.

```
for(j=0;j<n;j++)
        A[i][j]=0;
```

B.

```
for(i=0;i<n;i++)
        for(j=0;j<n;j++)
                A[i][j]=0;
```

6. Jaki jest czas działania instrukcji warunkowych:

```
if (<warunek>)
   <blok-if>
else
   <blok-else>
```

gdzie bloki `if` i `else` charakteryzują się ograniczeniami czasu działania odpowiednio na poziomie $f(n)$ i $g(n)$ [6].

7. Weźmy fragment programu:

```
small=i;
for(j=i+1 ; j<n ; j++)
        if(A[j]<A[small])
                small=j;
temp=A[small];
A[small]=A[i];
A[i]=temp;
```

Policzyć czas działania programu [7].

8. Znaleźć czas działania programu [8]:

```
i=0;
while(x!=A[i])
        i++;
```

9. Jaki jest prosty sposób na uniknięcie w poprzednim zadaniu błędu w razie, gdy `A[]` nie zawiera elementu `x`, bez konieczności znacznego zwiększania czasu działania pętli? [9]

10. Pokazać, w jaki sposób można określić czas działania poniższego programu metodą analizy drzewa struktury [10]:

```
for(i=0;i<n-1;i++)
        {
        small=i;
        for(j=i+1 ; j<n ; j++)
        if(A[j]<A[small])
                small=j;
        temp=A[small];
        A[small]=A[i];
        A[i]=temp;
        }
```

Literatura

[1] A.V. Aho, J.D. Ullman, *Wykłady z informatyki z przykładami w języku C*, przykład 3.1

[2] j.w., zadanie 3.3.1

[3] j.w., przykład 3.8

[4] j.w., przykład 3.9

[5] j.w., przykład 3.13

[6] j.w., podrozdział *Czas działania instrukcji warunkowych*

[7] j.w., przykład 3.17

[8] j.w., przykład 3.18

[9] j.w., ramka *Programowanie bardziej defensywne*

[10] j.w., podrozdział *Określanie czasu działania za pomocą analizy drzewa struktury*