Algorithm 1

1: **procedure** PROCEDURA1($n, x, a[\,]$)
2: **Input**
3: n — liczba całkowita, $n \geqslant 0$
4: x — liczba rzeczywista
5: a[ ] — zbiór/tablica/lista (n + 1) wartości rzeczywistych
6: **Output**
7: s — wartość rzeczywista
8: $s := 0.0$
9: **for** $i := n$ **to** 0 **do**
10: $p := a[n-1]$
11: **for** $j := i$ **to** 0 **do**
12: $p := p * x$
13: $s := s + p$
14: **return** s

Algorithm 2

1: **procedure** PROCEDURA2($n, x, a[\,]$)
2: **Input**
3: n — liczba całkowita, $n \geqslant 0$
4: x — liczba rzeczywista
5: a[ ] — zbiór/tablica/lista (n + 1) wartości rzeczywistych
6: **Output**
7: s — wartość rzeczywista
8: $s := a[0]$
9: **for** $i := n-1$ **to** 0 **do**
10: $s := s * x + a[n-i]$
11: **return** s

Zad. 1.
Odpowiedzieć na pytania:

1. Do czego służy algorytm zrealizowany w procedurze o nazwie "PROCEDURA1"? Odpowiedzieć szczegółowo.

   Odp.: Procedura służy do wyznaczenia wartości wielomianu $f(x)$ dla wartości x, przy czym w szczegółach realizowane sa nastepujace działania.

   (a) Stopień wielomianu wynosi n+1 - linia 9 zawiera petle, która wykonuje sie od wartości n do wartości 0, a wiec n+1 razy. Linia 11 zawiera petle, która wykonuje sie od i do 0, a wiec $i+1$ razy. Przy pierwszym obrocie petli zewnetrznej $i = n$ stad petla wewnetrzna wykona sie

$i + 1 = n + 1$ razy. Tyle razy wykona sie mnożenie wartości x w linii 12 stad najwyższa potega x w wielomianie to $n + 1$ (pierwsze mnożenie x przez wartość $a[n-1]$, każde kolejne przez x).

(b) Petla wewnetrzna z linii 11 zawsze wykona sie przynajmniej jeden raz. Dla $i = 0$ zmienna j ma sie zmieniać od 0 do 0, a zatem możliwy jest jeden obrót petli. Oznacza to, że najniższa potega zmiennej x w wielomianie jest równa 1.

(c) Z powodu aktualizacji wartości zmiennej p w linii 10 zawsze ta sama wartościa z listy a (spod indeksu $n-1$), wszystkie potegi x wymnażane sa zawsze przez ta sama wartość $(a[n-1])$.

(d) Ostateczna postać obliczanego wielomianu:

$$f(x) = a[n-1]x^{n+1} + a[n-1]x^{n} + a[n-1]x^{n-1} + \ldots + a[n-1]x$$

(e) Wywołanie procedury może prowadzić do błedu, gdy użytkownik wywoła procedure z wartościa $n = 0$ dopuszczalna przez warunki zadania. W jezyku Python indeks równy -1 nie powoduje błedu.

2. Do czego służy algorytm zrealizowany w procedurze o nazwie "PROCEDURA2"? Odpowiedzieć szczegółowo.

   Odp.: Procedura służy do wyznaczenia wartości wielomianu $f(x)$ dla wartości x, przy czym w szczegółach realizowane sa nastepujace działania.

   (a) Stopień wielomianu wynosi n - linia 9 zawiera petle, która wykonuje sie od wartości $n-1$ do wartości 0, a wiec n razy. Oznacza to, że wartość x bedzie wymnażana maksymalnie n razy.

   (b) Algorytm nazywany jest schematem Hornera.

   (c) W przeciwieństwie do procedury 1 wykorzystywane sa wszystkie wartości z tabeli a. Decyduje o tym dodanie wartości z tabeli a spod indeksu $n-i$ w linii 10 w każdym obrocie petli przy zmieniajacej sie wartości i.

   (d) Ostateczna postać obliczanego wielomianu:

   $$f(x) = a[0]x^{n} + a[1]x^{n-1} + a[2]x^{n-2} + \ldots + a[n]$$

3. Przy założeniu, że obie procedury zostana uruchomione dokładnie w tych samych warunkach, która procedura wykona sie w krótszym czasie? Dlaczego?

   Odp.:

   (a) W procedurze 1 sa dwie petle. Petla zewnetrzna wykona sie $n+1$ razy. Dla wartosci $i = n$ petla wewnetrzna rowniez wykona sie $n + 1$ razy. W najbardziej niekorzystnym przypadku liczba operacji mnożenia bedzie zatem rzedu n^2.

   (b) W procedurze 1 beda dodatkowo wykonane $n+1$ operacje dodawania.

(c) W procedurze 2 w petli z linii 9 (wykonujacej n obrotów) w każdym obrocie bedzie wykonane jedno mnożenie i jedno sumowanie. Zatem mamy n mnożeń i n sumowań. Liczba mnożeń jest rzedu n. To decyduje o szybszym wykonaniu procedury 2 niż procedury 1.

4. Oszacować ile raz dłużej bedzie wykonywać sie procedura o dłuższym czasie wykonania od procedury o krótszym czasie wykonania, przy założeniu, że operacje arytmetyczne dodawania i mnożenia trwaja tyle samo czasu. Przedstawić proces dochodzenia do wyniku szacowania.

   Odp.:

   (a) Dokładna liczbe operacji mnożenia w procedurze 1 można wyznaczyć jako sume ciagu arytmetycznego ze skokiem równym 1 (tak zmieniaja sie zmienne licznikowe petli). Pierwsza liczba mnożeń wyznaczona jest potega x w ostatnim wyrazie obliczanego wielomianu (x^1), a ostatnia liczba mnożeń wyznaczona jest potega pierwszego wyrazu obliczanego wielomianu x^{n+1}. Pośrednie potegi wyrazów wielomianu zmieniaja sie co 1. Możemy zatem wykorzystać ogólny wzór na sume b wyrazów ciagu arytmetycznego. W naszym przypadku, aby porównać obie procedury należy znormalizować b i uwzglednić, że wielomian z procedury 1 jest rzedu $n+1$ stad $b = n+1$.

   $$\text{Liczba mnożeń} = S_b = b \times \frac{c_1 + c_b}{2},$$

   gdzie $c_1 = 1, c_b = n+1$ i $b = n+1$. (W procedurze 1 przekazujemy parametr n, ale stopień wielomianu jest równy $c_b = n+1$ i interesuja nas mnożenia dotyczace $b = n+1$ wyrazów wielomianu, w których potega x zmienia sie od $c_1 = 1$ do $c_b = n+1$ - patrz odpowiedź 1d.) Stad liczba mnożeń w procedurze 1 równa sie:

   $$S_{n+1} = (n+1) \times \frac{n+2}{2}$$

   (b) Liczba sumowań w procedurze 1 równa sie $n+1$.

   (c) Liczba operacji sumowań jak i mnożeń w procedurze 2 równa sie n.

   (d) Załóżmy, że czas wykonania operacji jednego sumowania równa sie t. Z warunków zadania operacja mnożenia również trwa t.

   (e) Szacowanie czasu wykonania dłuższej procedury wzgledem krótszej:

   $$T_1 = (n+1)t + \left[(n+1) \times \frac{n+2}{2}\right] t$$

   $$T_2 = nt + nt$$

   $$T_1/T_2 = \left\{(n+1)t + \left[(n+1) \times \frac{n+2}{2}\right] t\right\} / (nt + nt) =$$

   $$= \left[(n+1) \times \left(1 + \frac{n+2}{2}\right)\right] /2n = \frac{(n+1)(n+4)}{4n}$$

5. Oszacować ile raz dłużej bedzie wykonywać sie procedura o dłuższym czasie wykonania od procedury o krótszym czasie wykonania, przy założeniu, że operacja mnożenia trwa tyle samo czasu co dwie operacje dodawania. Przedstawić proces dochodzenia do wyniku szacowania.

   Odp.:

   (a) Założenia tak same jak poprzednio oprócz czasu operacji mnożenia. Obecnie operacja mnożenia trwa $2t$.

   (b) Szacowanie czasu wykonania dłuższej procedury wzgledem krótszej:

$$T_1 = (n+1)t + \left[(n+1) \times \frac{n+2}{2}\right] 2t$$

$$T_2 = nt + n2t$$

$$T_1/T_2 = \left\{(n+1)t + \left[(n+1) \times \frac{n+2}{2}\right] 2t\right\} / (nt + 2nt) =$$

$$= [(n+1) + (n+1) \times (n+2)] / 3n = \frac{(n+1)(n+3)}{3n}$$

   Przykładowo, dla $n = 10$ procedura 1 bedzie wykonywała sie 3,8 razy dłużej niż procedura 2, gdy operacje sumowania i mnożenia trwaja tyle samo czasu (na przykład procedura numer 2 zwróci wynik po 10 sekundach, podczas, gdy procedura 1 dopiero po 38 sekundach) i 4,76 razy dłużej, gdy operacja mnożenia jest wykonywana dwa razy dłużej niż sumowanie. Należy zwrócić uwage, że chociaż złożoność czasowa procedury 1 zależy od kwadratu wartości n, a procedury 2 zależy liniowo od n, to nie obserwujemy wprost zależności $T_1 = nT_2$. Za pierwszym razem (równe czasy operacji arytmetycznych) bedzie to ok. $n/4$ razy dłużej, a za drugim ok. $n/3$ razy.

Zad. 2.

1. Sformułować pytania ogólne, na jakie może znaleźć odpowiedź elektrotechnik odpowiednio korzystajacy z przedstawionej procedury o nazwie "PROCEDURAEL".

   A
   - Gdy switch równa sie True - Jaka jest czestotliwość rezonansowa w obwodzie RLC dla zadanych parametrów R, L i C?
   - Gdy switch równa sie False - Jakie jest przesuniecie fazowe (kat wyrażony w stopniach) pradu wzgledem napiecia w obwodzie RLC dla zadanych parametrów R, L, C i f?

   B
   - Gdy switch równa sie True - Jaka jest indukcyjność cewki w obwodzie rezonansowym RLC dla zadanych parametrów R, C i f?
   - Gdy switch równa sie False - Jakie jest przesuniecie fazowe (kat wyrażony w radianach) pradu wzgledem napiecia w obwodzie RLC dla zadanych parametrów R, L, C i f?

C • Gdy switch równa sie True - Jaka jest pojemność kondensatora w obwodzie rezonansowym RLC dla zadanych parametrów R, L i f?

- Gdy switch równa sie False - Jakie jest przesuniecie fazowe (kat wyrażony w radianach) pradu wzgledem napiecia w obwodzie RLC dla zadanych parametrów R, L, C i f?

D • Gdy switch równa sie True - Jaka jest zawada w obwodzie RLC dla zadanych parametrów R, L, C i f?

- Gdy switch równa sie False - Jakie jest przesuniecie fazowe (kat wyrażony w stopniach) pradu wzgledem napiecia w obwodzie RLC dla zadanych parametrów R, L, C i f?

2. Zakładajac, że I to liczba liter w imieniu studenta, N liczba liter w nazwisku studenta, a A, to suma dwóch ostatnich cyfr w albumie studenta, wyznaczyć wynik działania procedury PROCEDURAEL wywołanej z nastepujacymi parametrami: $R = I*10$, $L = N*10^{-5}$, $C = (A+1)*10^{-6}$, $f = (I + N + A) * 10^3$ i switch równe True oraz False. Podać wielkości fizyczne, dla których wyznaczono wartość oraz sama wartość z jednostka główna z układu SI.

   Odp.: Wyniki indywidualne uzależnione od parametrów I, N, A. Sprawdzenie poprawności wykonywania działań i obsługi nawiasów.

3. Napisać program w jezyku Python, który pozwoli uzyskać wyniki, o których mowa w podpunkcie 2 tego zadania.

   Odp.: Program wprost przeniesiony z pseudokodu. Trudność polega na znajomości załaczenia pakietu math instrukcja `import math` oraz odpowiednie wykorzystanie stałej `math.pi` oraz funkcji `math.atan()` i `math.sqrt()`. Wynik poprawnie napisanego programu może być podstawa odpowiedzi na pytanie 2. Jednostki, których należało użyć:

   - R - opór elektryczny czynny - Ohm
   - L - indukcyjność - Henr
   - C - pojemność elektryczna - Farad
   - f - czestotliwość (również obliczana jako wynik) - Hz (lub 1/s)
   - kat - obliczany był wynik - w grupie A i D stopień, w grupie B i C - radian
   - zawada (moduł impedancji) - obliczany był wynik - Ohm

Algorithm 3

procedure ProceduraELA($R, L, C, f, switch$)
    Input
        R liczba rzeczywista różna od 0
        L liczba rzeczywista
        C liczba rzeczywista
        f liczba rzeczywista
        switch wartość typu logicznego
    Output
        w wartość rzeczywista
    if switch = True **then**
        $w := 1.0/(2 * PI * sqrt(LC))$
        return w
    else
        $w := (arctg(2 * PI * f * L - ((1.0/(2 * PI * f * C)))/R)) * 180/PI$
    return w

Algorithm 4

procedure ProceduraELB($R, L, C, f, switch$)
    Input
        R liczba rzeczywista różna od 0
        L liczba rzeczywista
        C liczba rzeczywista
        f liczba rzeczywista
        switch wartość typu logicznego
    Output
        w wartość rzeczywista
    if switch = True **then**
        $w := 1.0/(4 * PI * PI * f * f * C)$
        return w
    else
        $w := arctg(2 * PI * f * L - ((1.0/(2 * PI * f * C)))/R)$
    return w

Algorithm 5

```
procedure PROCEDURAELC(R, L, C, f, switch)
    Input
        R               liczba rzeczywista różna od 0
        L               liczba rzeczywista
        C               liczba rzeczywista
        f               liczba rzeczywista
        switch          wartość typu logicznego
    Output
        w               wartość rzeczywista
    if switch = True then
        w := 1.0/(4 * PI * PI * f * f * L)
        return w
    else
        w := arctg(2 * PI * f * L − ((1.0/(2 * PI * f * C)))/R)
    return w
```

Algorithm 6

```
procedure PROCEDURAELD(R, L, C, f, switch)
    Input
        R               liczba rzeczywista różna od 0
        L               liczba rzeczywista
        C               liczba rzeczywista
        f               liczba rzeczywista
        switch          wartość typu logicznego
    Output
        w               wartość rzeczywista
    if switch = True then
        p := 2 * PI * f * L − (1.0/(2 * PI * f * C))
        w := sqrt(R * R + p * p)
        return w
    else
        w := (arctg(2 * PI * f * L − ((1.0/(2 * PI * f * C)))/R)) * 180/PI
    return w
```