Lab 4b (7 pkt) dynamiczne tablice oraz struktury

W zadaniu zdefiniowano strukturę **`array`** odpowiadającą reprezentacji dynamicznej tablicy **`n`** wartości całkowitych.
Gdy tablica reprezentuje tablicę współczynników **wielomianu**, to element o najwyższym indeksie odpowiada współczynnikowi przy najwyższej potędze, a **`w[0]`** – wyraz wolny.

Zadanie dotyczy wybranych działań na **wielomianach** (stopnia>=1), które są reprezentowane za pomocą struktury **`array`**

$$w(x) = x^n + \cdots + w_1 x + w_0$$

Funkcja **main** jest przygotowana „**prawie” w całości** i nie należy jej modyfikować (oczywiście poza zakomentowaniem / odkomentowaniem testowanych fragmentów), **wyjątkiem** jest ETAP 3, gdzie należy **dopisać** fragment dla wywołania funkcji.
Wszystkie etapy należy wykonać w podanej kolejności, w których należy zaimplementować wszystkie **niezbędne funkcje**.

Etap 1 (3 pkt= 1 + 1/2 + 1,5)

Należy zaimplementować **3 funkcje**, które **można wywołać** następująco:

```
k=polynomial_generator(&w, 3, a, b);
```

- Funkcja **`polynomial_generator`** wypełnia strukturę **`w`** tworząc wewnątrz dynamiczną tablicę o rozmiarze **`3+1`** (**`s=3`**-stopień wielomianu), współcz. przy najwyższej potędze to zawsze **1**, współcz. o **indeksie 0** to zawsze p^s, gdzie **p** jest liczba całkowita wylosowana z **`[a,b]`**, ale **różna** od **`0`** wg wzoru: **`a+rand()%(b-a+1)`**
  Pozostałe współczynniki wielomianu są równe 0.
  Zwraca w przypadku niepowodzenia 0, wpp zwraca 1.

```
print(tablica);
```

- Funkcja **`print`** wypisuje na ekran elementy 1-wymiarowej tablicy **`w`** o rozmiarze **`n`** zawartej w strukturze **`tab`** (w kolejności od maksymalnego indeksu do indeksu 0)
  dla przykładu :
  `[ 1  0  0  -125 ]`

```
k= multiplication(w1, w2, &w3);
```

- Funkcja **`multiplication`** na podstawie struktur wielomianów **`w1`** oraz **`w2`** znajduje współczynniki iloczynu wielomianów w strukturze **`w3`**
  Funkcję należy zaimplementować ogólnie, tzn. dla dowolnych wielomianów.
  Zwraca w przypadku niepowodzenia 0, wpp zwraca 1.

Etap 2 (2 pkt = 1 + 1)

Należy zaimplementować **2 funkcje**, które można **wywołać** następująco:

```
z=value(&w, x);
```

- Funkcja **`value`** oblicza wartość wielomianu przechowywanego w strukturze **`w`** dla argumentu **`x`**.
  Zwraca obliczoną wartość wielomianu.
  Zastosuj algorytm **Hornera**

`int k=find_roots(w, &roots)`

- Funkcja **`find_roots`** szuka listy **całkowitych pierwiastków** wielomianu. Całkowite pierwiastki są dzielnikami całkowitymi (np. 1,-1, itd.) współczynnika wolnego o indeksie 0 (zakładamy, że jest on niezerowy)
  Zwraca w przypadku niepowodzenia 0, wpp zwraca 1.

Etap 3 (2 pkt = 1 + 1)

Należy zaimplementować **2 funkcje**, które można **wywołać** następująco:

`k=polyder( w, &w_prim );`

Funkcja **`polyder`** - na podstawie wielomianu **`w`** - oblicza współczynniki wielomianu pochodnej i wypełnia odpowiednio strukturę **`w_prim`**. Stopień wielomianu będącego pochodną jest o 1 niższy od stopnia pierwotnego wielomianu.
Zwraca w przypadku niepowodzenia 0, wpp zwraca 1.

`k=single_root(w, r);`

- Funkcja **`single_root`** – na podstawie struktury **`w`** – zwraca 1, gdy **r** jest to pierwiastek **jednokrotny**, wpp zwraca 0.
  Wewnątrz funkcji wykorzystaj funkcję **`polyder`**, a następnie sprawdź zerowanie pochodnej dla podanej wartości **r**.

Wyniki dla przykładu testowego

```
ETAP1-----------------------------------------------------------------
Wylosowano wsp. dla wielomianu w1:
[ 1  0  0  -125  ]
Wylosowano wsp. dla wielomianu w2:
[ 1  0  0  0  0  243  ]
Wsp. wielomianu w3=w1*w2 w bazie potegowej:
[ 1  0  0  -125  0  243  0  0  -30375  ]



ETAP2-----------------------------------------------------------------

Znaleziona lista calkowitych pierwiastkow dla w3[ 5  -3  ]

Sprawdzenie, czy to zera wielomianu?
dla x=r0=-3 w(x)= 0
dla x=r1= 5 w(x)= 0

ETAP3-----------------------------------------------------------------
Wsp. pochodnej wielomianu:
[ 8  0  0  -625  0  729  0  0  ]

Czy pojedyncze zero wielomianu? - przez sprawdzenie wartosci pochodnej:
```

```
 zero pojedyncze: r0= -3
 zero pojedyncze: r1=  5

KONIEC
Press any key to close this window . . .
```