Zestaw 6

Poniższe zadania służą przećwiczeniu zagadnień związanych z obiektami funkcyjnymi i wyrażeniami lambda, również algorytmami i kontenerami. Proszę precyzyjnie implementować zapisane treści.

- 1. Zadanie polegać będzie na utworzeniu wektora i posortowaniu go, według poniższych specyfikacji:
 - a. Utwórz wektor od typu int o nazwie vec, z zaalokowanymi 100 miejscami
 - b. Algorytmem iota wypełnij go wartościami od 1 do 100
 - c. Za pomocą algorytmu shuffle wymieszaj zawartość, wypisz (dowolnie) na ekran
 - d. Algorytmem sort oraz odpowiednim predykatem jako argument, posortuj zawartość od największej do najmniejszej wartości, wypisz na ekran
 - e. Napisz obiekt funkcyjny (użyj strukturę), który posortuje jak wyżej, zastosuj i wypisz na ekran
 - f. Posortuj wektor vec za pomocą wyrażenia lambda, ale uwaga: należy napisać takie "nietypowe" kryterium, że liczby zaczynające się od 1 są najpierw, potem od 2... i tak dalej. Czyli efektywnie powinno z tego wyjść: 1 10 11 12 13 14 15 16 17 18 19 100 2 20 21...
- 2. Napisz wyrażenie lambda, które wywołane potem z jednym argumentem wyliczy wartość wskazanego elementu ciągu Fibonacciego, ale uwaga, nie chodzi o znane rozwiązanie, tylko zewnętrzne wyrażenie: auto fibo = ma mieć zagnieżdżoną definicję kolejnego wyrażenia, np. auto inner_fibo = ... o argumentach int oraz const auto& impl, gdzie w środku będzie wywołane już "klasyczne" wyliczenie, rekurencyjne wołające impl, zaś poniżej tej definicji return inner_fibo(n, inner_fibo); (czyli inner_fibo będzie właśnie argumentem impl). Następnie wypisać w pętli for jakieś przykładowe kolejne wyrazy ciągu.
- 3. Utwórz obiekt map<int, string> mis, który zainicjalizuj od razu parami {0, "zero"}, {1, "jeden"}, ..., {9, "dziewiec"}. Utwórz wektor z liczbami całkowitymi (można też wykorzystać ten z zadania 1), następnie za pomocą algorytmu transform, przejdź po tym wektorze (pierwszy i drugi argument) tak, że wyrażenie lambda (czwarty argument) niech zwraca dla danej liczby z wektora odpowiadającą jej nazwę z mapy, która to nazwa zostanie wysłana na ekran za pomocą iteratora strumienia wyjściowego (trzeci argument).
- **4.** W zadaniu tym wykorzystany będzie algorytm accumulate, którego argumenty są następujące: iterator_początek, iterator_koniec, wartość początkowa, binarny_obiekt_funkcyjny. Ostatni argument domyślnie jest sumą, wtedy za wartość początkową należy podać 0. Jeśli chcemy policzyć iloczyn, to za wartość początkową należy podać 1 oraz wstawić odpowiedni obiekt funkcyjny (multiplies). Proszę wykonać następujące kroki:
 - a. Utworzyć obiekt vector<int>, wypełniony lub zainicjalizowany kilkunastoma liczbami
 - b. Napisać wyrażenie lambda (uchwycone przez auto nazwa) takie, że przyjmuje dwa argumenty, string oraz int, i zwraca sklejony z nich string, na zasadzie pierwszy argument + "-" + przekonwertowany drugi argument (przypominam o funkcji to string).
 - c. Za pomocą accumulate przejść po całym wektorze, podając jako wartość początkową (przekonwertowany do string) pierwszy element tego wektora, a jako wielkość wywoływaną powyższe wyrażenie lambda. Chodzi o to, żeby np. z wektora 1,2,3,4 otrzymać w ten sposób string: 1-2-3-4. Proszę go zapisać pod auto s = i wypisać s na ekran. Wskazówka: funkcja next pozwala przejść od pierwszego elementu do kolejnego, przyda się dla pierwszego argumentu accumulate.
- 5. Utworzyć list<char> i zainicjalizować dokładnie takimi elementami: { 'a', 'c', 'b', 'd', 'f', 'e', 'h', 'g'}. Napisać wyrażenie lambda (uchwycone pod jakąś nazwę za pomocą auto), które będzie przyjmować dwa argumenty, każdy niech będzie const auto&, a zwracać bool, na zasadzie decydowania czy dana literka | (lewa) jest "mniejsza niż" literka p (prawa), według kolejności jak

w powyższej liście. Czyli, na przykład wyrażenie to dla argumentów 'c' i 'b' powinno zwrócić true, bo na liście 'c' jest na lewo od 'b'. Następnie utworzyć obiekt set na typie char, zadając mu jako kryterium sortowania właśnie powyższe wyrażenie lambda, oraz (koniecznie) inicjalizując następującym zestawem: {'a', 'a', 'g', 'c', 'b', 'f', 'h', 'c', 'e', 'd', 'e'} (proszę sprawdzić jak się tworzy obiekt set wraz z inicjalizacją i kryterium sortowania). Wynik wypisać: for (auto c: mySet) cout << c; (powinno wyjść acbdfehg).