Ćwiczenie 3 z Podstaw programowania.
Język C++, programy pisane w nieobiektowym stylu programowania

Zofia Kruczkiewicz

Zakres

- Podstawowe algorytmy przetwarzania tablic (wypełnianie, porównywanie elementów, wyszukiwanie, przesuwanie, usuwanie, dodawanie elementów)
- Tablica pseudo-dynamiczna (statyczna tablica z licznikiem wykorzystywanych elementów)
- Parametryzacja algorytmów
- Dobór sposobu przekazywania parametrów do/z funkcji

Zad. 1 - Zasięg deklaracji i widoczność zmiennych (obowiązkowe)

Napisz w komentarzach, jaką wartość mają poszczególne zmienne i gdzie znajdują się ich definicje.

```
int a;
void fun();
void main()
  {
    int a;
    a=8;
    fun();
}
void fun()
{    a=6;
     int a;
     a=2;
      {  int a, b;
              a=9;
              b=89;
      }
      a=20;
      //b=7;
}
```

Zad2 (obowiązkowe)

1. Napisz program, który w opcjach wykonuje następujące czynności na tablicy zdefiniowanej jako ***int tab[N],*** gdzie na początku liczba elementów w tablicy jest równa ***ile=0***:
   1.1. wstawia elementy typu **int** do tablicy, wypełniając całą tablicę. Po wykonaniu tej opcji liczba elementów powinna być równa ***ile=N***.
   1.2. wyznacza wartość maksymalną i wyświetla ją oraz indeks tego elementu, jeśli tablica zawiera dane
   1.3. wyznacza wartość minimalną i wyświetla ją oraz indeks tego elementu, jeśli tablica zawiera dane
   1.4. wyznacza wartość średnią i wyświetla ją oraz indeks tego elementu, jeśli tablica zawiera dane
   1.5. wyświetla zawartość tablicy, jeśli zawiera dane.

Uwagi:

- Należy wprowadzić zmienną ile, która przechowuje liczbę elementów tablicy. Wartość ***ile*** równa zero świadczy o tym, ze tablica jest pusta. Wartość różna od zera oznacza, że w tablicy są dane. Nie może ona przekroczyć wartości ***N***. Oznacza to, że jeżeli ***ile=N***, wtedy tablica jest pełna. Wartość N musi być zadana programie, np. **const int N=10;**
- Należy dodać komentarze w kodzie programu wyjaśniające działanie danego fragmentu programu reprezentującego działanie danego algorytmu odpowiednio opisanego w p. 1.1-1.5.

Zad3 (obowiązkowe)

2. Napisz program, który w opcjach wykonuje następujące czynności na tablicy zdefiniowanej jako ***int tab[N],*** gdzie na początku liczba elementów w tablicy jest równa ***ile=0***:

   2.1. po każdym wywołaniu tej opcji wstawia po jednym elemencie typu **int** do tablicy, zawsze za ostatnio wstawionym elementem, czyli indeks tej pozycji jest równy ile. Należy zawsze sprawdzić przed wprowadzeniem, czy liczba elementów ***ile*** jest mniejsza od wartości ***N***. Po każdym wprowadzeniu danej należy zwiększyć liczbę elementów ***ile o 1***

   2.2. wyszukuje w tablicy element o wartości równej wartości podanej z klawiatury . Jeśli znajdzie taki element wyświetla jego wartość oraz wartość jego indeks u.

   2.3 usuwa element z tablicy, którego wartość jest równa wartości podanej z klawiatury, jeśli tablica zawiera dane. Usuwanie polega na zsuwaniu elementów tablicy tzn. jeśli znaleziono element w tablicy o indeksie równym ***j***, wtedy należy wykonać w pętli kolejno: ***tab[j] = tab[j+1],*** gdzie wartość ***j+1 <ile*** i po zakończeniu pętli zmniejszyć wartość ***ile*** o ***1.*** Jeżeli ***j = ile-1***, wtedy wystarczy zmniejszyć liczbę elementów ***ile*** o ***1***

   2.4. wyświetla zawartość tablicy, jeśli zawiera dane.

Uwagi:

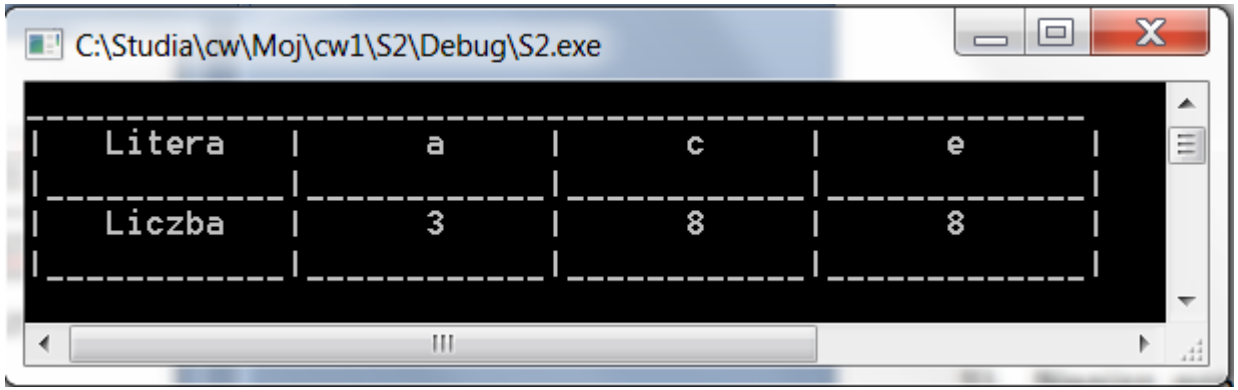
1. Należy wprowadzić zmienną ***ile***, która przechowuje liczbę elementów tablicy. Wartość ***ile*** równa zero świadczy o tym, ze tablica jest pusta. Wartość różna od zera oznacza, że w tablicy są dane. Nie może ona przekroczyć wartości ***N.*** Wartość ***ile=N*** oznacza, ze tablica jest pełna. Wartość N musi być zadana programie, np. **const int N=10;**
2. Można wykonać funkcje realizujące czynności przedstawione w podpunktach 2.1-2.4. Należy zaproponować sposób przekazywania parametrów przez funkcje.
3. Należy dodać komentarze w kodzie programu wyjaśniające działanie danego fragmentu programu reprezentującego działanie danego algorytmu odpowiednio opisanego w p. 2.1-2.4.

Zad4 (dodatkowe)

1) Napisz program, który będzie wczytywał i analizował tekst (ciąg liter) wprowadzany z klawiatury aż do momentu naciśnięcia klawisza ‘k’.
2) Jeśli we wprowadzanym tekście wystąpią litery: a, c, e należy wyświetlić na ekranie histogram występowania tych liter alfabetu np.

Litera	**a**	**c**	**e**
Liczba	5	0	1

3) Liczbę litery **a** należy przechowywać w pierwszym elemencie tablicy, **c** w drugim elemencie tablicy, a **e** w trzecim elemencie
4) Należy dodać komentarze w kodzie programu wyjaśniające działanie danego fragmentu programu reprezentującego działanie danego algorytmu odpowiednio opisanego w p. 1-3.



Zad5 (dodatkowe)

Algorytm wyszukania liczb pierwszych metodą sita Eratostenesa. Należy wyznaczyć wszystkie liczby pierwsze w podzbiorze liczb naturalnych {1..N} za pomocą algorytmu sita Eratostenesa wg podanego algorytmu. Należy wykonać schemat blokowy i program z użyciem tablicy ***int tab[N].***

1) Utworzyć tablicę zawierającą N elementów i wstawić do każdego elementu wartość 0

   1.1) Podaj ***N*** z klawiatury

   1.2) Jeśli ***N<2***, powtórz krok 1.1

   1.3) Ustaw ***i:=2***

   1.4) Dopóki *i<=N* wykonuj kolejne kroki, w przeciwnym wypadku przejdź do kroku 2

      1.4.1) wstaw 0 do elementu tablicy o indeksie *i*

      1.4.2) zwiększ ***i:=i+1*** i przejdź do **kroku 1.4.**

2) Zakłada sie, że pewne indeksy elementów są szukanymi liczbami pierwszymi i po zakończeniu algorytmu elementy tablicy o tych indeksach będą zawierać wartość 0, natomiast pozostałe elementy mają wartość 1, ponieważ nie są liczbami pierwszymi. Stąd należy wstawić na początku wartość 1 do elementu o indeksie równym 1, ponieważ 1 nie jest liczba pierwszą.

3) Ustawić ***ost_Liczp:=1;***

4) Na podstawie faktu, że każda liczba złożona nie większa niż *N ma dzielnik nie większy niż sqrt(N),* wykonuj kolejne kroki, gdy ***ost_Liczp*ost_Liczp <=N****, w przeciwnym wypadku przejdź do kroku 5:*

4.1) Należy zwiększyć ***ost_Liczp*** *o 1:* ***ost_Liczp:=ost_Liczp+1***

4.2) **Wykonuj kolejne kroki , jeśli jest prawdziwy warunek *ost_Liczp<=N) and (tab[ost_Liczp]=1*, w przeciwnym wypadku przejdź do kroku 4.3.**

4.2.1) zwiększaj ***ost_Liczp*** *o 1:* ***ost_Liczp:=ost_Liczp+1*** *(poszukiwanie kolejnej liczby pierwszej, czyli elementu tablicy o indeksie* ***ost_Liczp*** *nie zawierającej wartości 1)*

4.4.2) przejdź do **kroku 4.2**

4.3) Należy wyznaczyć podwojoną wartość ***ost_Liczp ost*** i wyznaczyć numer ***i*** kolejnej liczby, która nie jest liczbą pierwszą : ***i:= ost_Liczp*2*** *(rozpoczęcie kolejnego etapu wykreślania liczb, które nie są liczbami pierwszymi)*

4.4) Dopóki *i<=N*, *wykonaj w kolejnych krokach eliminacje liczb, które nie są liczbami pierwszymi,* ponieważ są ich wielokrotnościami, w przeciwnym wypadku przejdź do kroku 4.

4.4.1) wstaw wartość ***1*** to elementu tablicy o wierszu równym ***i*** : ***tab[i]:=1***

4.4.2) dodaj wartość ***ost_Liczp*** *do* ***i: i:=i+ost_Liczp****, następnie przejdź do* ***kroku 4.4***

5) Wyświetl zawartość tablicy na ekranie:

5.1) wstaw ***i:=1***

5.2) dopóki *i<=N* *wykonuj kolejne kroki, w przeciwnym wypadku zakończ algorytm*

5.2.1) jeśli ***tab[i]=0****, wyświetl indeks elementu jako wartość kolejnej liczby pierwszej*

5.2.2) wyznacz kolejny indeks ***i:=i+1*** *i przejdź do* ***kroku 5.2.***

Wyjaśnienie:

Wyrażenie : ***ost_Liczp:=ost_Liczp+1*** oznacza, że zmiennej ***ost_Liczp*** została przypisana wartośc o 1 większa od poprzedniej