27 listopada 2020

Michał Odrobina

Algorytmy i struktury danych

Projekt I, grupa 6

Spis treści

Odwołania do schematów blokowych:

[(1) *dowolny_ciag( )*]

[(2) *sprawdz_k( )*]

[(3) *przelicz_ciag( )*]

Wstęp wraz z opisem problemu

Celem projektu jest stworzenie programu dotyczącego zagadnień ciągu arytmetycznego. Ciąg ten ma mieć zastosowanie tylko dla liczb naturalnych. Głównym zadaniem programu jest znalezienie ciągu, którego suma będzie większa od zadanej liczby ‘k’. Jednakże jego długość ma być najkrótsza z możliwych. W przypadku gdy suma całego ciągu będzie większa lub równa od zadanej liczby ‘k’, ma pojawić się komunikat informujący o braku podtablic spełniających zadane warunki. Program również dopuszcza użycia jednego wyrazu ciągu w przypadku, gdy będzie on większy od zadanej liczby ‘k’.

Opis podstaw teoretycznych zagadnienia

Przy tworzeniu algorytmu wyliczającego ciąg trzeba było uwzględnić podstawowy wzór na *n*-ty wyraz ciągu arytmetycznego (a_n) o pierwszym wyrazie a_1 i różnicy r:

$$a_n = a_1 + (n-1)r$$

Zastosować także odpowiedni algorytm umożliwiający zsumowanie całego ciągu i porównanie go z liczbą ‘k’.

Sam algorytm wyliczający najkrótszą sumę ciągu większą od ‘k’ opiera się o fakt dodawania kolejnych wyrazów ciągu do poprzedniej sumy, w sposób zapętlony. Każda suma zostaje przyrównana do ‘k’ i jeśli jest ona większa od zadanej liczby, przyjmuje ją jako wartość poprawną, większą od ‘k’. W momencie, w którym znany jest wynik sumy, zostaje on rozbity na poszczególne wyrazy ciągu. Całość, wyrazy ciągu wraz z wynikiem zostają wyświetlone na wyjściu oraz przesłane do utworzonego pliku tekstowego „Wyniki.txt”.

Opis szczegółów implementacji problemu

Podstawą było stworzenie funkcji odpowiedzialnych za wyliczenie samego ciągu arytmetycznego. Przy tworzeniu funkcji należało uwzględnić podanie wartości pierwszego wyrazu ciągu, ilości wyrazów w ciągu oraz różnice ciągu. A także zastosować wzór na *n*-ty wyraz ciągu arytmetycznego. W momencie gdy użytkownik wpisze wartość ujemną lub równą 0, otrzymujemy on komunikat o błędzie i wartość należy podać od nowa.

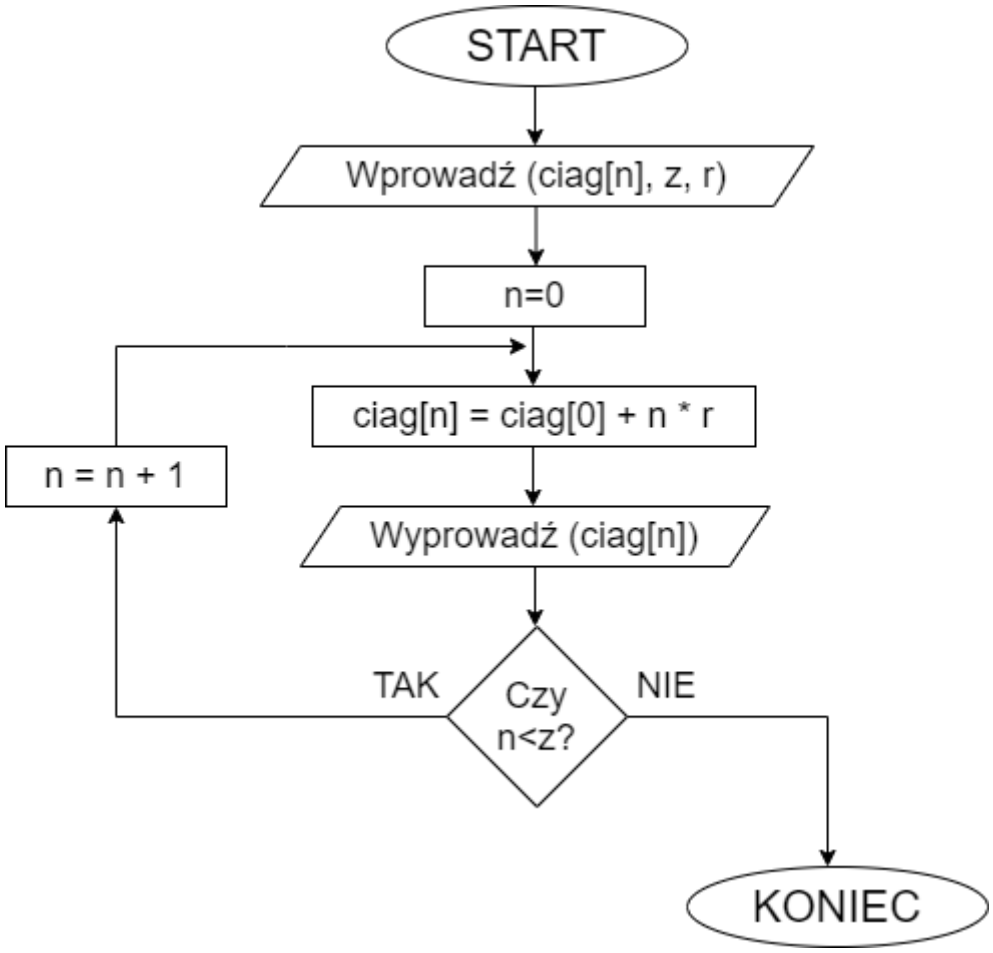
Funkcji sprawdzającej czy przypadkiem suma całego ciągu nie jest mniejsza od ‘k’. Polega ona na dodawaniu kolejnego wyrazu ciągu do sumy poprzednich. Po czym następuje porównanie całości z liczbą ‘k’. W przypadku gdy suma jest mniejsza, następuje wyświetlenie komunikatu i powrót do Menu Wyboru.

A także głównej funkcji zadania, odpowiedzialnej za porównanie z zadaną liczbą ‘k’ każdego wyrazu/ każdej sumy ciągu, a w przypadku gdy ta była mniejsza od ‘k’ dodania do poprzedniego wyrazu/sumy kolejnego wyrazu ciągu (n+1). W momencie gdy wynik porównania jest większy zadanej liczby, zostaje on podany dalej na wyjście i rozłożony na poszczególne wyrazy ciągu wchodzące w skład tej sumy. Suma wraz wyrazami zostają wyświetlone, a także przesłane do pliku tekstowego „Wyniki.txt”.

Wszystkie funkcje zostały umieszczone w Menu Wyboru wraz z funkcją umożliwiającą wyczyszczenie zawartości pliku „Wyniki.txt”.

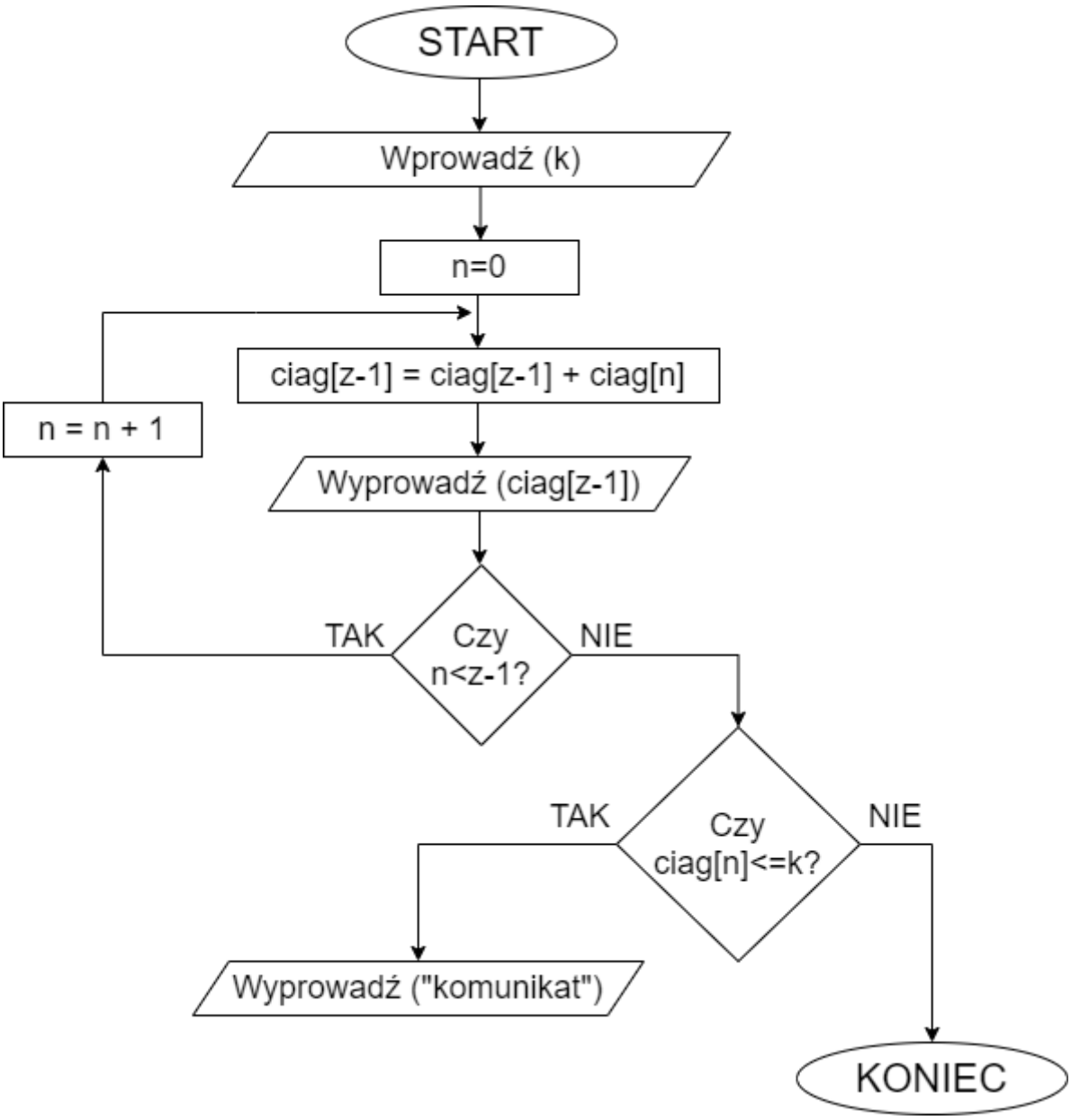
Schemat blokowy algorytmu

(1) Schemat funkcji: *dowolny_ciag( )*



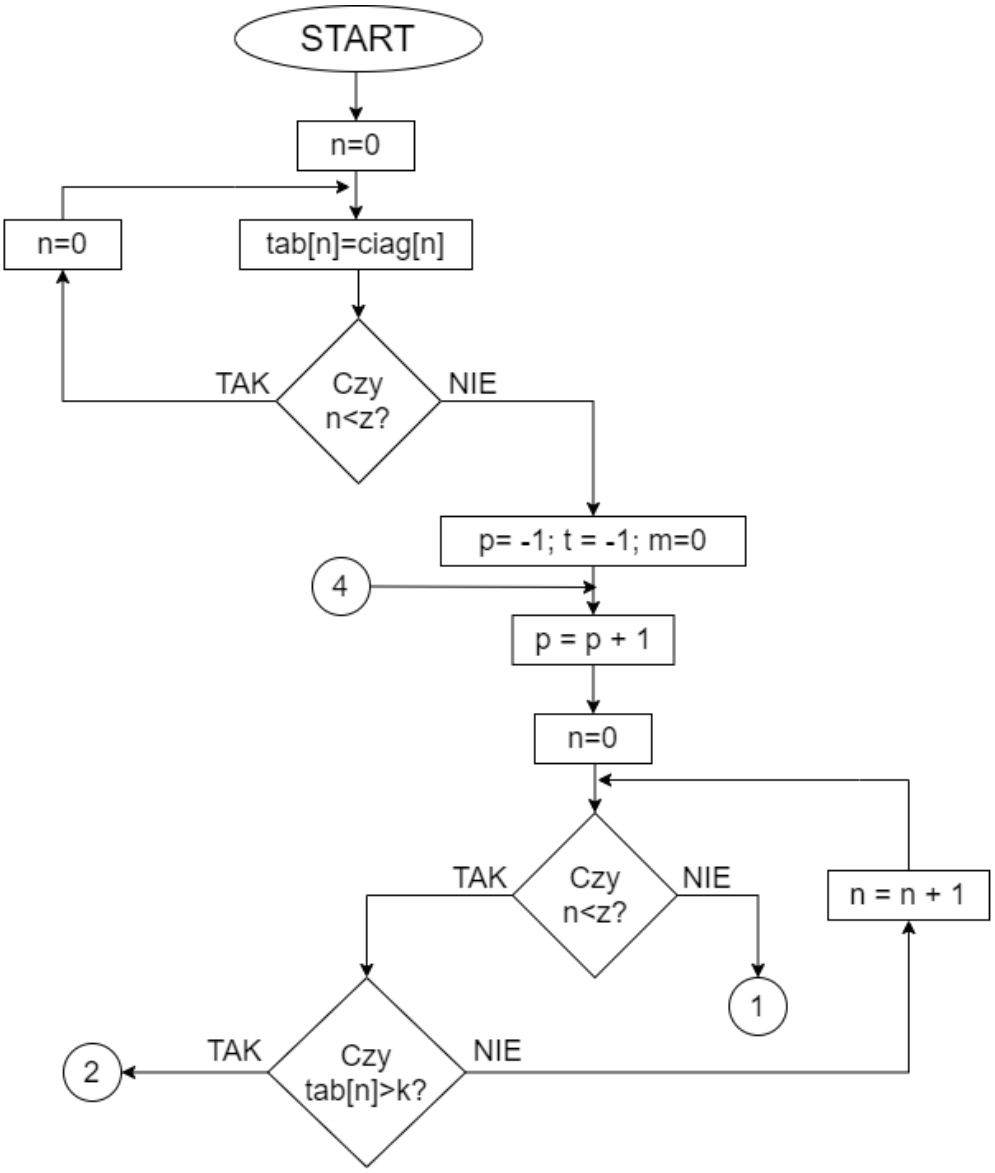
Schemat blokowy 1

(2) Schemat funkcji: *sprawdz_k( )*

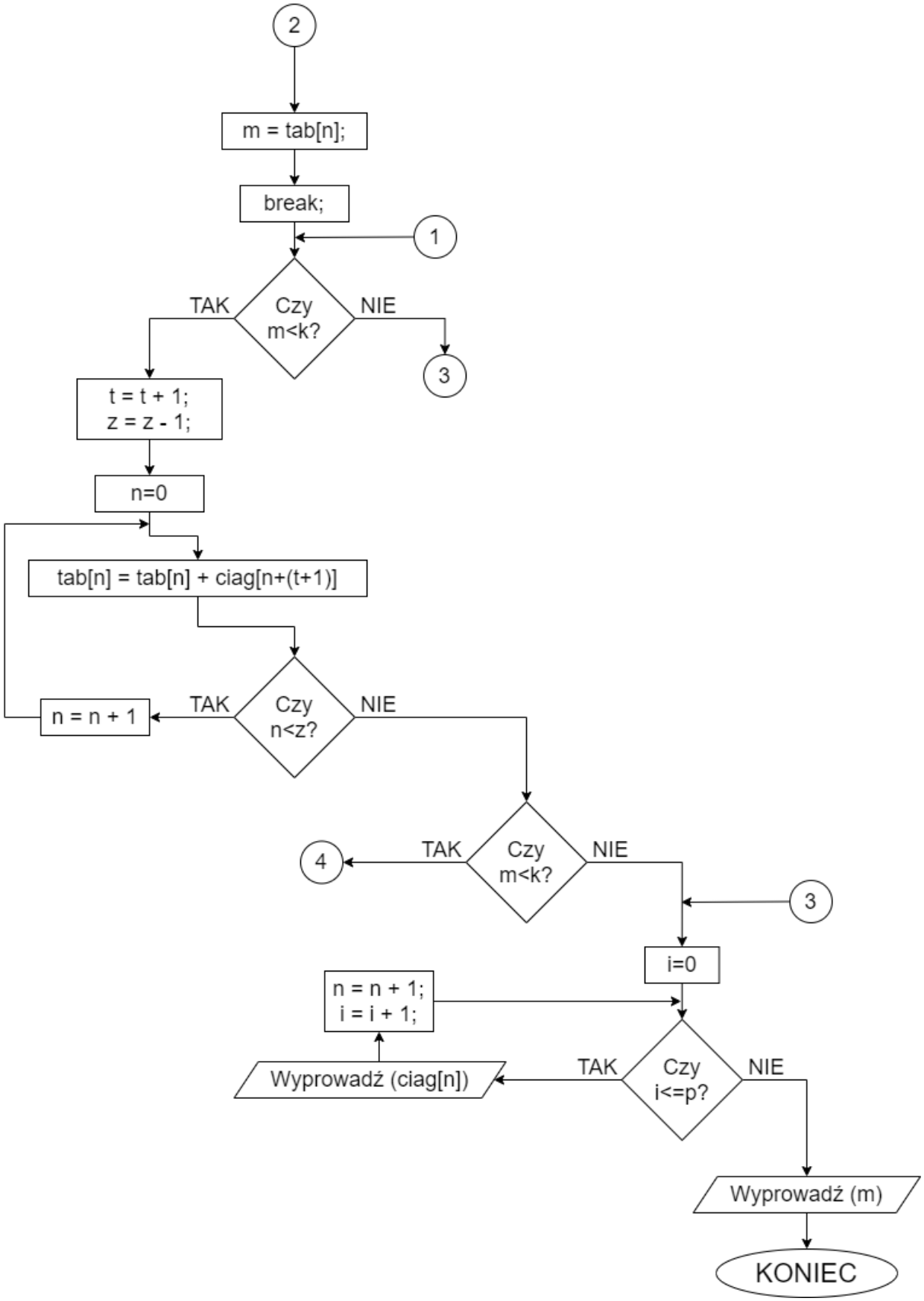


Schemat blokowy 2

(3) Schemat funkcji: *przelicz_ciag( )*



Schemat blokowy 3 cz. 1



Schemat blokowy 4 cz. 2

Pseudokod głównego algorytmu, *przelicz_ciag( )*

Przyrównaj wszystkie wyrazy ciągu (ciag[n]) do tab[n]

Sprawdź dla każdego wyrazu ciągu czy jest większy od 'k'

Jeżeli NIE, to przejdź dalej

Jeżeli TAK, to przyrównaj ten ciąg do 'm'

Dla 'm' mniejszego od 'k'

Dla 'm' większego od 'k'

Dodaj kolejny wyraz ciągu do poprzedniego

Powtórz całą operację, aż jeden z wyrazów ciągu będzie równy 'm'

Wyświetl wszystkie wyrazy sumy ciągu większego od 'k'

Wyświetl sumę ciągu większego od 'k'

Rezultaty testów programu

1) W Menu Wyboru:
   - Wpisując znaki inne niż te, które zostały zaproponowane w Menu Wyboru, wyświetla się komunikat o błędzie.
   - Wpisując wiele znaków następujących po sobie, program dla każdego z osobna wyświetla komunikat o błędzie. Kiedy wszystkie komunikaty znikną program wraca do pierwotnego stanu wyboru.

a) Wybór ‘1’:
- Pozwala przejść do zadania dla ciągu od 1 do 8 o różnicy 1. Wyświetla wartości ciągu a_1, n oraz r oraz wszystkie wyrazy tego ciągu. Po czym należy podać liczbę ‘k’.

b) Wybór ‘2’:
- Pozwala przejść do zadania dla ciągu naturalnego dowolnego, podanego przez użytkownika. W przypadku podania liczby ujemnej, wyświetla się komunikat nakazujący wpisanie tylko liczby dodatniej. W momencie wpisania ciągu znaków lub liczby zmiennoprzecinkowej powstaje błąd zawarty w podpunkcie c)*. Wyświetla wartości ciągu a_1, n oraz r oraz wszystkie wyrazy tego ciągu. Po czym należy podać liczbę ‘k’.

c) Dla wyboru ‘1’ oraz ‘2’, podanie liczby ‘k’:
- W przypadku podania liczby całkowitej program podaje na wyjście odpowiednio poprawny wynik sumy i wyrazy ciągu tej sumy. W przypadku wpisania liczby zmiennoprzecinkowej, program „wyrzuca” nas do Menu Wyboru. *Jednakże po wpisaniu ciągu znaków program ponownie „wyrzuca” nas do Menu Wyboru, w momencie ponownego wybrania opcji ‘1’ lub ‘2’, nie pozwala na poprawne użytkowanie funkcji. Wyświetla natychmiast komunikat o przejściu do Menu Wyboru za pomocą klawisza ENTER. Program powraca do prawidłowego działania po jego ponownym uruchomieniu.

d) Wybór ‘3’:
- Wyświetla komunikat o wyczyszczeniu pliku „Wyniki.txt” i czyści plik.

e) Wybór ‘4’:
- Zatrzymuje program.

Wnioski i podsumowanie

Program jest wstanie wyliczać sumę najkrótszego ciągu większego od zadanej liczby ‘k’. Jest on jednak ograniczony ze względu na liczbę bajtów. Maksymalna liczba jaką jest w stanie przyjąć to 2147483647. Ma problem także z odczytem ciągów znaków zamiast liczb. Jednakże podając poprawne dane na wejście otrzymujemy oczekiwane wyniki, które są następnie zapisywane w pliku tekstowym. Samo zadanie, do którego należało stworzyć algorytm, wymusiło na mnie zastosowanie wielu podejść do tego problemu. Aby rozwiązać zadany problem zdałem sobie sprawę jak ważne jest przeanalizowanie każdej linijki krok po kroku. Stworzyłem w tym celu osoby kod o nazwie „Test algorytmu”. Dzięki takiemu podejściu byłem wstanie przejść dalej i zrozumieć w pełni jak działa program.

Appendix

A: main( )

```
int main()
{


   char nr_wyboru;

   cout << endl << "Witaj!" << endl << endl;
   Sleep(1000);


   do
   {
      system("cls");


      cout << "-----------------------------------------------------------------" << endl;

      cout << "Program oblicza sume najkrotszego ciagu liczb naturalnych, wiekszych od zadanej liczby 'k'." << endl;

      cout << "-----------------------------------------------------------------" << endl;

      cout << "MENU WYBORU" << endl << endl;

      cout << "Zatwierdzanie w MENU WYBORU odbywa sie poprzez wcisniecie odpowiedniego klawisza." << endl << endl;
      cout << "Wybierz '1': Dla zadanego ciagu od 1 do 8, roznica = 1." << endl;
      cout << "Wybierz '2': Dla dowolnego wlasnego ciagu." << endl;
      cout << "Wybierz '3': Aby wymazac zawartosc pliku \"Wyniki.txt\"." << endl;
      cout << "Wybierz '4': Aby zamknac program." << endl << endl;


      cout << "Wybierz opcje: ";
      nr_wyboru = getch();
      cout << nr_wyboru;
      Sleep(500);
```

```
    // MENU WYBORU


     switch(nr_wyboru)
     {
     case '1':
        ciag1_8();
        sprawdz_k();
        if(ciag[n]<=k)
        {
           break;
        }
        przelicz_ciag();
        break;
     case '2':
        dowolny_ciag();
        sprawdz_k();
        if(ciag[n]<=k)
        {
           break;
        }
        przelicz_ciag();
        break;
     case '3':
        wymaz_txt();
        break;
     case '4':
        break;
     default:
        cout << endl << endl << "Blad ! Mozesz wybrac tylko opcje podane wyzej..." << endl;
        Sleep(3000);
        break;

     }
  } while(nr_wyboru!='4');


  return 0;
}
```

B: wymaz_txt( )

```
void wymaz_txt()
{
   ofstream oout("Wyniki.txt");
   oout << "";
   cout << endl << endl <<  "Wymazano zawartosc pliku \"Wyniki.txt\" !...";

   Sleep(2000);
}
```

C: ciag1_8( )

```
void ciag1_8()
{

   system("cls");


   ciag[0]=1;
   z=8;
   int r=1;



      cout << "Pierwszy wyraz ciagu: " << ciag[0] << endl;
      cout << "Ilosc liczb w ciagu: " << z << endl;
      cout << "Roznica ciagu: " << r << endl;



   cout << "Oto liczby ciagu: " << endl;
   out << endl << "Dla liczb ciagu: ";

   for(n=0;n<z;n++)
   {
      ciag[n] = ciag[0] + n * r;
      cout << ciag[n] << " ";
      out << ciag[n] << " ";
   }

   cout << endl;

   n = 0;

}
```

D: dowolny_ciag( )

```
void dowolny_ciag()
{
   system("cls");

   int r;


   do
   {


      cout << "Podaj pierwszy wyraz ciagu: ";
      cin >> ciag[0];
      if(ciag[0]<=0)
      {
         cout << "Liczba musi byc dodatnia !" << endl;
         Sleep(1000);
         system("cls");
      }

   } while(ciag[0]<=0);


   do
   {

      cout << "Podaj ilosc liczb w ciagu: ";
      cin >> z;
      if(z<=0)
      {
         cout << "Liczba musi byc dodatnia !" << endl;
         Sleep(1000);
         system("cls");
         cout << "Podaj pierwszy wyraz ciagu: " << ciag[0] << endl;
      }

   } while(z<=0);
```

```
    do
    {
       cout << "Podaj roznice ciagu: ";
       cin >> r;
       if(r<=0)
       {
          cout << "Liczba musi byc dodatnia !" << endl;
          Sleep(1000);
          system("cls");
          cout << "Podaj pierwszy wyraz ciagu: " << ciag[0] << endl;
          cout << "Podaj ilosc liczb w ciagu: " << z << endl;
       }

    } while(r<=0);


    cout << "Oto liczby ciagu: " << endl;
    out << endl << "Dla liczb ciagu: ";

    for(n=0;n<z;n++)
    {
       ciag[n] = ciag[0] + n * r;
       cout << ciag[n] << " ";
       out << ciag[n] << " ";
    }

    cout << endl;

    n = 0;

}
```

E: sprawdz_k( )

```
void sprawdz_k()
{
    cout << "Podaj k: ";
    cin >> k;


    out << endl << "Dla k: " << k << endl << endl;
    cout << endl;


    e = ciag[z-1];
   for(n=0;n<z-1;n++)
    {
        ciag[z-1] += ciag[n];
    }



    if(ciag[n]<=k)
    {
        cout << "Brak podtablic spelniajacych zadane warunki." << endl;
        cout << endl << endl << "Nacisnij ENTER, aby wrocic do MENU WYBORU...";
        getchar();
        getchar();

    }



}
```

F: przelicz_ciag( )

```
void przelicz_ciag()
{

   ciag[z-1] = e;

  for(n=0;n<z;n++)
   {
      tab[n] = ciag[n];
   }

   cout << endl;

  cout << "------------------------------------------------------------" << endl;

   p = -1;
   t = -1;
   m = 0;

   do
   {
      p++;

      for(n=0;n<z;n++)
      {

         if(tab[n]>k)
         {
            m=tab[n];
           break;
         }
      }

      if(m<k)
      {

         t++;
         z -= 1;

         for(n=0;n<z;n++)
         {
            tab[n] += ciag[n+(t+1)];

         }
      }
   } while(m<k);
```

```
    cout << "Wyjscie: ";
    out << "Wyjscie: ";
    for(int i=0;i<=p;i++)
    {

        cout << ciag[n] << " ";
        out << ciag[n] << " ";
        n++;

    }



    cout << endl << "Suma ciagu wyjsciowego: " << m;
    out << endl << "Suma ciagu wyjsciowego: " << m << endl << endl;
    out << "-----------------------------------------" << endl << endl;


    cout << endl << endl << "Nacisnij ENTER, aby wrocic do MENU WYBORU...";
    getchar();
    getchar();



}
```