Generowanie liczb pseudolosowych w języku C++

- Wprowadzenie
- Przeczytaj
- Symulacja interaktywna
- Sprawdź się
- Dla nauczyciela

Źródło: Riho Kroll, domena publiczna.

Liczby losowe oraz pseudolosowe mają wiele zastosowań praktycznych, poznaliśmy je w e-materiale Liczby pseudolosowe. Programy służące do generowania takich liczb mogą być wykorzystywane loteriach, grach losowych czy przy tworzeniu modeli matematycznych.

Z tego e-materiału dowiesz się, jak przy pomocy języka C++ wygenerować liczby pseudolosowe.

Ciekawi cię, jak wyglądają implementacje algorytmu generowania liczb pseudolosowych w innych językach programowania? Możesz się z nimi zapoznać w dwóch pozostałych e-materiałach z tej serii:

- Generowanie liczb pseudolosowych w języku Java,
- Generowanie liczb pseudolosowych w języku Python.

Więcej zadań? Sięgnij do Generowanie liczb pseudolosowych – zadania maturalne.

> **Twoje cele**
>
> - Przeanalizujesz, jak generować liczby pseudolosowe w języku C++.
> - Napiszesz program w języku C++, który wygeneruje tyle liczb losowych, ile zechce użytkownik, i wypisze je na ekranie.

- Wykonasz kilka ćwiczeń, w których sprawdzisz swoje umiejętności w zakresie generowania liczb pseudolosowych.

Przeczytaj

Jak wygenerować liczbę?

Do wygenerowania liczby w języku programowania C++ używamy funkcji `rand()`. Generuje ona liczbę z zakresu od 0 do `RAND_MAX`, gdzie `RAND_MAX` to stała zapisana w bibliotece, do której należy funkcja `rand()` – czyli jest to `<cstdlib>`.

Napiszmy krótki program, który przy pomocy funkcji `rand()` wygeneruje liczbę z zakresu od 0 do `RAND_MAX`.

Zacznijmy od napisania szkieletu głównej funkcji programu – `main()`, a w niej zadeklarowania zmiennej typu całkowitego `int` o nazwie `wylosowana`. Będziemy przechowywać w niej wylosowaną liczbę.

```
#include <iostream>

int main() {

    int wylosowana;

    return 0;
}
```

Następnie użyjmy funkcji `std::rand()` do wylosowania liczby. Wynik losowania zapiszmy w zmiennej `wylosowana`.

> **Ważne!**
>
> W związku z tym, że używasz funkcji `rand()`, pamiętaj o dodaniu odpowiedniej biblioteki:
>
> ```
> #include <cstdlib>
> ```

```
#include <iostream>
#include <cstdlib>

int main() {

    int wylosowana = std::rand();
```

```
7
8       return 0;
9   }
```

Ostatnie, co musimy zrobić, to wypisać wylosowaną liczbę. Robimy to przy pomocy strumienia `std::cout`.

```
1   #include <iostream>
2   #include <cstdlib>
3
4
5   int main() {
6
7       int wylosowana = std::rand();
8
9       std::cout << wylosowana;
10
11      return 0;
12  }
```

Po kilkukrotnym uruchomieniu programu zauważymy, że program za każdym razem losuje te same liczby. Dlaczego tak się dzieje?

Przy starcie programu ziarno, czyli wartość, od której algorytm zaczyna generowanie kolejnych liczb, jest stałe i niezmienne. Stan ten może ulec zmianie pod warunkiem, że na początku programu w miejsce tej wartości wprowadzi się nową wartość i będzie ona inna przy każdorazowym uruchomieniu programu.

Dlatego, aby program nie losował tych samych liczb, użyjemy poniższej instrukcji:

```
1   srand(time(NULL));
```

Ta linijka powoduje, że ziarno generatora za każdym razem startuje od innej wartości, co sprawia, że wylosowane liczby po każdym uruchomieniu programu są inne. Warto zaznaczyć, że funkcja ta w swoim działaniu wykorzystuje czas systemowy.

Aby funkcja działała prawidłowo, należy dodać odpowiednią bibliotekę, w której ta funkcja się znajduje.

```
1   #include <time.h>
```

Dodajmy powyższe dwie linijki do wcześniej napisanego programu i sprawdźmy rezultat.

```
#include <iostream>
#include <cstdlib>
#include <time.h>

int main() {
    srand(time(NULL));

    int wylosowana = std::rand();

    std::cout << wylosowana;

    return 0;
}
```

Teraz losowana liczba za każdym razem jest inna.

Generowanie liczb z danego zakresu

Wiemy już, jak wygenerować liczbę pseudolosową, zatem teraz napiszmy program, który wygeneruje liczby z danego zakresu. Kontynuujmy pracę z wcześniej napisanym kodem.

Poniższa linijka kodu do zmiennej `wylosowana` wygeneruje liczbę z zakresu od 1 do 48.

```
int wylosowana = (std::rand() % 48) + 1;
```

Liczba 1 oznacza najmniejszą możliwą do wylosowania liczbę, a wartość po modulo (%) to wielkość przedziału, do którego należą liczby całkowite, jakie możemy wylosować.

Przeanalizujmy poniższe przykłady kodu:

```
int wylosowana = (std::rand() % 48) + 7;
//losuje liczby z przedziału <7, 54>
```

```
int wylosowana = (std::rand() % 100) + 2;
//losuje liczby z przedziału <2, 101>
```

```
int wylosowana = (std::rand() % 150) + 50;
//losuje liczby z przedziału <50, 199>
```

Z powyższych linijek kodu możemy teraz wyprowadzić wzór na przedział losowanych liczb.

Jeżeli oznaczymy zmienne w następujący sposób:

```
int wylosowana = (std::rand() % LP) + LS;
```

Wówczas liczby będą losowane z przedziału:

$$\langle LS,\ LS + LP - 1 \rangle$$

gdzie LS to liczba startowa, a LP – liczebność przedziału

> **Praca domowa**
>
> Napisz program, który wygeneruje liczbę z przedziału $\langle 77,\ 169 \rangle$.

Przykład programu w języku C++

Napiszmy program, który dla podanego przez użytkownika przedziału będzie losować 100 liczb, a po wylosowaniu wartości granicznej wypisze wyraz „INFORMATYKA".

Zacznijmy od zadeklarowania dwóch zmiennych, do których zapiszemy granice przedziału, jakie użytkownik poda z klawiatury. Będą to:

```
int L_granica;
int P_granica;
```

Kod, dzięki któremu użytkownik poda własne granice przedziału, będzie wyglądał tak:

```
std::cout << "Podaj lewą granicę przedziału" << std::endl;
std::cin >> L_granica;
std::cout << "Podaj prawą granicę przedziału" << std::endl;
std::cin >> P_granica;
```

Następnym krokiem jest napisanie pętli, która wykona się 100 razy.

```
for(int i = 0; i < 100; i++) {

}
```

Teraz napiszemy kod, który wygeneruje liczbę pseudolosową z zadanego przez użytkownika zakresu, zapisze ją w zmiennej `wylosowana` i wypisze na standardowym wyjściu.

```
int wylosowana = (std::rand() % (P_granica - L_granica + 1)) + L_

std::cout << wylosowana << std::endl;
```

Aby program przy każdym uruchomieniu losował inne liczby, musimy pamiętać o dodaniu:

```
srand(time(NULL));
```

> **Ważne!**
>
> W związku z użyciem powyższych funkcji wbudowanych, nie zapomnijmy o dodaniu odpowiednich bibliotek:
>
> ```
> #include <cstdlib>
> #include <time.h>
> ```

Program jest prawie gotowy, dodajmy jeszcze tylko warunek, że jeżeli wylosowana liczba jest liczbą graniczną, to wypisuje się napis „INFORMATYKA”.

```
if (wylosowana == L_granica || wylosowana == P_granica) {
    std::cout << "INFORMATYKA" << std::endl;
}
```

Zobaczmy, jak wygląda kod całego programu:

```
#include <iostream>
#include <cstdlib>
#include <time.h>

int main() {
```

```
6
7       srand(time(NULL));
8
9       int L_granica;
10      int P_granica;
11
12      std::cout << "Podaj lewą granicę przedziału" << std::endl;
13      std::cin >> L_granica;
14      std::cout << "Podaj prawą granicę przedziału" << std::endl;
15      std::cin >> P_granica;
16
17      for(int i = 0; i < 100; i++) {
18
19          int wylosowana = (std::rand() % (P_granica - L_granica +
20
21          std::cout << wylosowana << std::endl;
22
23          if (wylosowana == L_granica || wylosowana == P_granica) {
24              std::cout << "INFORMATYKA" << std::endl;
25          }
26      }
27
28      return 0;
29  }
```

Słownik

> **funkcja wbudowana**
>
> funkcja, która została już zaimplementowana; należy do jednej z bibliotek lub klas; przykładami takich funkcji w języku C++ są: `rand()`, `srand()`, `length()`

Symulacja interaktywna

Polecenie 1

Napisz program, który umożliwi użytkownikowi wpisanie n liczb z określonego zakresu, a następnie wylosuje tyle samo liczb z tego samego zakresu i wypisze pierwszą, która znajduje się w obu tablicach. Przetestuj jego działanie dla sześciu liczb z przedziału $\langle 1,\ 48 \rangle$.

Specyfikacja problemu:

Dane:

- n – liczba pobieranych liczb; liczba naturalna
- `los_max` – maksymalna dozwolna wylosowana wartość
- `wpisane` – tablica liczb naturalnych
- `wylosowane` – tablica liczb wylosowanych przez program; tablica liczb naturalnych

Wynik:

Na standardowym wyjściu program prezentuje pierwszą liczbę występującą w obu tablicach.

```
#include <iostream>
#include <cstdlib>
#include <time.h>

int main() {

    srand(time(NULL));


    int wpisane[6];
    int wylosowane[6];

    // tutaj dopisz kod

```

```
1
```

Polecenie 2

Porównaj swoje rozwiązanie z prezentacją.

1

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Naszym zadaniem jest napisanie programu, który umożliwi użytkownikowi wpisanie sześciu liczb z zakresu $\langle 1,\ 48 \rangle$, następnie wylosuje sześć liczb z przedziału $\langle 1,\ 48 \rangle$ i wypisze pierwszą liczbę, którą użytkownik „trafił”.

2

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Zacznijmy od zbudowania szkieletu programu, czyli zadeklarowania głównej funkcji programu - `main`. To w niej będzie znajdować się pozostała część naszego programu.

```
#include <iostream>

int main() {

    return 0;
}
```

3

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

W kolejnym kroku, do zmiennej n, pobieramy od użytkownika liczbę liczb, które zostaną pobrane; zmienna ta będzie również określać liczbę liczb wylosowanych przez program. Zapisujemy do zmiennej `losMax` największą wartość jaką program może wylosować.

```
#include <iostream>

int main() {

    std::cout << "Podaj liczbę przetwarzanych liczb:" << std::endl;
    int n = 0;
    std::cin >> n;
    std::cout << "Podaj największą wartość losowanej liczby:" << std::endl;
    int losMax = 0;
    std::cin >> losMax;

```

```
    return 0;
}
```

4

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Następnie zadeklarujmy dwie tablice:

- `int wpisane[n]` – do tej tablicy typu całkowitego będziemy zapisywać w pętli `for` kolejne wpisane przez użytkownika liczby;
- `int wylosowane[n]` – do tej tablicy typu całkowitego będziemy zapisywać w pętli `for` wylosowane liczby.

```
#include <iostream>

int main() {

    std::cout << "Podaj liczbę przetwarzanych liczb:" << std::endl;
    int n = 0;
    std::cin >> n;
    std::cout << "Podaj największą wartość losowanej liczby:" << std::endl;
    int losMax = 0;
    std::cin >> losMax;


    int wpisane[n];
    int wylosowane[n];

    return 0;
}
```

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Napiszemy teraz pętlę `for`, w której będziemy pobierać od użytkownika kolejne liczby i zapisywać je w tablicy `wpisane[n]`.

Kolejne wartości pobieramy za pomocą strumienia `cin`.

```
1  #include <iostream>
2
3  int main() {
4
5          std::cout <<
   "Podaj liczbę
   przetwarzanych liczb:" <<
   std::endl;
6      int n = 0;
7      std::cin >> n;
8      std::cout << "Podaj
   największą wartość
   losowanej liczby:" <<
   std::endl;
9      int losMax = 0;
10     std::cin >> losMax;
11
12
13     int wpisane[n];
14     int wylosowane[n];
15
16     for(int i = 0; i < n;
   i++) {
17         std::cout <<
   "Podaj liczbę z zakresu
   <1, " << losMax << ">" <<
   std::endl;
18         std::cin >>
   wpisane[i];
19     }
20
```

```
    return 0;
}
```

6

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Kolejnym krokiem jest napisanie szkieletu pętli, w której będzie się odbywać losowanie liczb. Losujemy n liczb, więc zmienna sterująca przyjmuje wartości od 0 do n.

```
#include <iostream>

int main() {

    std::cout << "Podaj liczbę przetwarzanych liczb:" << std::endl;
    int n = 0;
    std::cin >> n;
    std::cout << "Podaj największą wartość losowanej liczby:" << std::endl;
    int losMax = 0;
    std::cin >> losMax;


    int wpisane[n];
    int wylosowane[n];

    for(int i = 0; i < n; i++) {
        std::cout << "Podaj liczbę z zakresu <1, " << losMax << ">" << std::endl;
        std::cin >> wpisane[i];
    }
```

```
20
21      for(int i = 0; i < n;
    i++) {
22
23      }
24
25      return 0;
26  }
```

7

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

1. Deklarujemy zmienną `wylosowana`, do której zapisujemy wylosowaną liczbę z przedziału $\langle 1, \texttt{losMax} \rangle$.
2. Zapisujemy ją do tablicy `wylosowane` w linii 25.
3. Wypisujemy wylosowaną liczbę za pomocą strumienia `cout` w linii 27.

```
1  #include <iostream>
2
3  int main() {
4
5      std::cout << "Podaj
   liczbę przetwarzanych
   liczb:" << std::endl;
6      int n = 0;
7      std::cin >> n;
8      std::cout << "Podaj
   największą wartość
   losowanej liczby:" <<
   std::endl;
9      int losMax = 0;
10     std::cin >> losMax;
11
12
13     int wpisane[n];
14     int wylosowane[n];
15
```

```
16      for(int i = 0; i < n;
    i++) {
17          std::cout <<
    "Podaj liczbę z zakresu
    <1, " << losMax << ">" <<
    std::endl;
18          std::cin >>
    wpisane[i];
19      }
20
21      for(int i = 0; i < n;
    i++) {
22
23          int wylosowana =
    (std::rand() % (losMax)) +
    1;
24
25          wylosowane[i] =
    wylosowana;
26
27          std::cout <<
    "Wylosowana liczba:" <<
    wylosowana << std::endl;
28
29      }
30
31      return 0;
32  }
```

8

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Aby program z każdym uruchomieniem losował inne liczby, dodajmy:

```
1  srand(time(NULL));
```

9

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

W związku z użyciem funkcji `rand()` oraz `srand()`, pamiętajmy o dodaniu odpowiednich bibliotek:

```
#include <cstdlib>
#include <time.h>
```

10

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

Jedną z ostatnich czynności, jakie musimy wykonać, jest napisanie kodu sprawdzającego, czy liczby podane przez użytkownika są równe wylosowanym. Potrzebujemy do tego dwóch zagnieżdżonych pętli.

```
#include <iostream>
#include <cstdlib>
#include <time.h>

int main() {

    srand(time(NULL));
    std::cout << "Podaj liczbę przetwarzanych liczb:" << std::endl;
    int n = 0;
    std::cin >> n;
    std::cout << "Podaj największą wartość losowanej liczby:" << std::endl;
    int losMax = 0;
    std::cin >> losMax;


    int wpisane[n];
```

```
17      int wylosowane[n];
18
19      for(int i = 0; i < n;
    i++) {
20          std::cout <<
    "Podaj liczbę z zakresu
    <1, " << losMax << ">" <<
    std::endl;
21          std::cin >>
    wpisane[i];
22      }
23
24      for(int i = 0; i < n;
    i++) {
25
26          int wylosowana =
    (std::rand() % (losMax)) +
    1;
27
28          wylosowane[i] =
    wylosowana;
29
30          std::cout <<
    "Wylosowana liczba:" <<
    wylosowana << std::endl;
31
32      }
33
34
35      for(int i = 0; i < n;
    i++) {
36          for(int j = 0; j <
    n; j++) {
37              if(wpisane[i]
    == wylosowane[j]) {
38
39              }
40          }
41      }
42
43      return 0;
44  }
```

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P117gmdzk

W przypadku znalezienia tych samych wartości w tablicach `wpisane` i `wylosowane` wypisujemy odpowiedni komunikat.

Oto kod całego programu:

```
#include <iostream>
#include <cstdlib>
#include <time.h>

int main() {

    srand(time(NULL));
    std::cout << "Podaj liczbę przetwarzanych liczb:" << std::endl;
    int n = 0;
    std::cin >> n;
    std::cout << "Podaj największą wartość losowanej liczby:" << std::endl;
    int losMax = 0;
    std::cin >> losMax;


    int wpisane[n];
    int wylosowane[n];

    for(int i = 0; i < n; i++) {
        std::cout << "Podaj liczbę z zakresu <1, " << losMax << ">" << std::endl;
        std::cin >> wpisane[i];
    }

```

```
24      for(int i = 0; i < n;
    i++) {
25
26          int wylosowana =
    (std::rand() % (losMax)) +
    1;
27
28          wylosowane[i] =
    wylosowana;
29
30          std::cout <<
    "Wylosowana liczba:" <<
    wylosowana << std::endl;
31
32      }
33
34      for(int i = 0; i < n;
    i++) {
35          for(int j = 0; j <
    n; j++) {
36              if(wpisane[i]
    == wylosowane[j]) {
37                  std::cout
    << "Pierwsza trafiona
    liczba: " << wpisane[i] <<
    std::endl;
38                  return 1;
39              }
40          }
41      }
42
43
44      return 0;
45  }
```

Polecenie 3

Symulacja bada właściwości przykładowego liniowego generatora liczb pseudolosowych, którego wzór podany jest poniżej. Podobne algorytmy mogą być stosowane w celu wyznaczania liczb losowych w metodach informatycznych. Modyfikując parametry generatora, obserwuj, w jaki sposób zmienia się rozkład generowanych przez niego liczb. Spróbuj dobrać parametry w taki sposób, aby rozkład był jak najbardziej jednorodny.

Zasób interaktywny dostępny pod adresem https://zpe.gov.pl/a/DkDhjLvhV

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, którego zadaniem będzie powtarzanie losowania do czasu wylosowania pewnej liczby.

Wskazówka: pamiętaj o dodaniu odpowiednich bibliotek. Zastanów się, czy losowany zakres wpływa na liczbę losowań.

Swój program przetestuj dla liczby 4.

Specyfikacja problemu:

Dane:

- `liczba` – wylosowana wartość; liczba naturalna

Wynik:

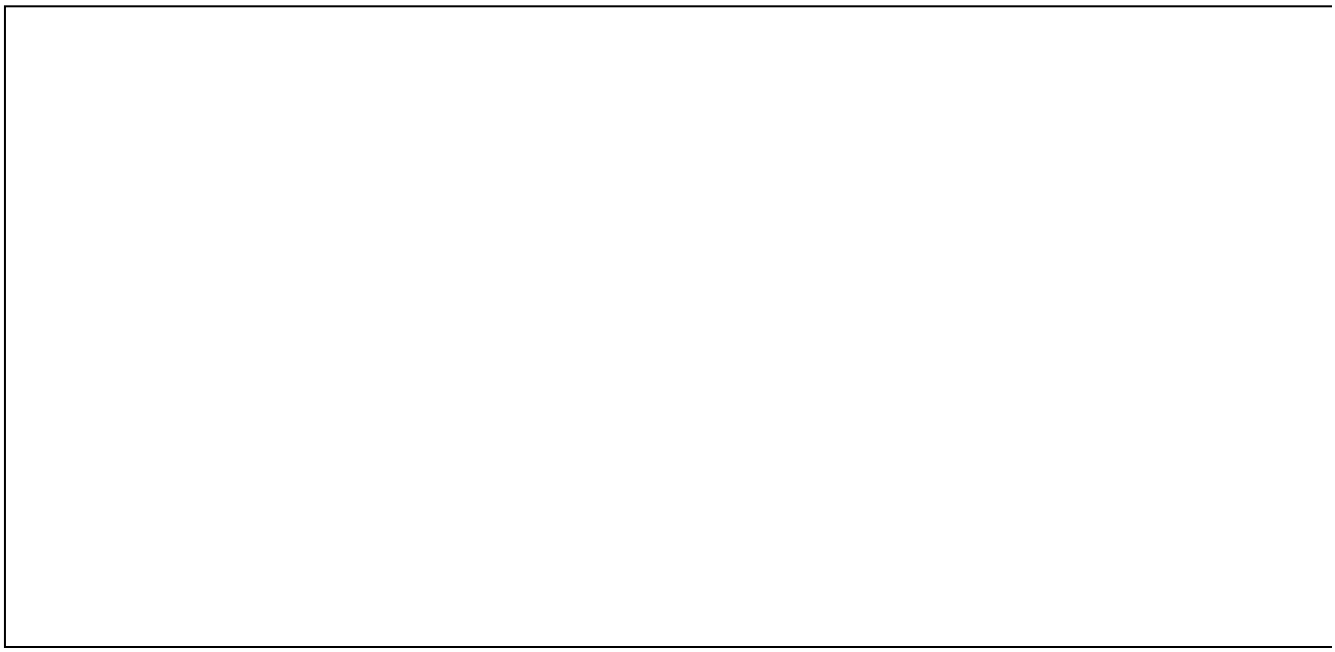
Po wylosowaniu danej liczby zostaje ona wypisana na standardowym wyjściu.

Twoje zadania

1. Program ma wylosować liczbę 4 i ją wypisać.

```cpp
#include <iostream>

int main() {

    int liczba = // Dopisz kod

    std::cout << liczba;

    return 0;
}
```

```
1
```



Ćwiczenie 2

Napisz program, który będzie losował liczby z zakresu $\langle 1;\ n \rangle$, a każde wylosowanie liczby x wypisze od nowej linii. Po trzecim losowaniu ma zakończyć działanie. Kod jest niepełny – uzupełnij go w odpowiednich miejscach.

Przetestuj działanie programu dla zakresu $\langle 1;\ 10 \rangle$ oraz liczby 7.

Specyfikacja problemu:

Dane:

- n – górny zakres losowania; liczba naturalna
- x – liczba, która zostanie wypisana na standardowym wyjściu po jej wylosowaniu

Wynik:

Na standardowym wyjściu program wypisje x, za każdym razem, gdy zostanie wylosowana.

Twoje zadania

1. Program ma generować kolejne liczby i wypisywać (od nowej linii), jeżeli wygeneruje liczbę 7. Po trzecim wypisaniu kończy działanie.

```
#include <iostream>
#include <cstdlib>
//brakujacy kod

int main() {

    srand(time(NULL));

    int ilosc = 0;

    do {


```

```
13          int wylosowana = (rand() % //brakujacy kod +
```

```
1
```

Ćwiczenie 3

Napisz program, który na pewno wylosuje pewną liczbę dziesięć razy i ją wypisze. Pamiętaj, aby dobrze dobrać zakres i dodać odpowiednie biblioteki.

Przetestuj jego działanie dla liczby 19.

Specyfikacja problemu:

Dane:

- `liczba` – wylosowana wartość; liczba naturalna

Wynik:

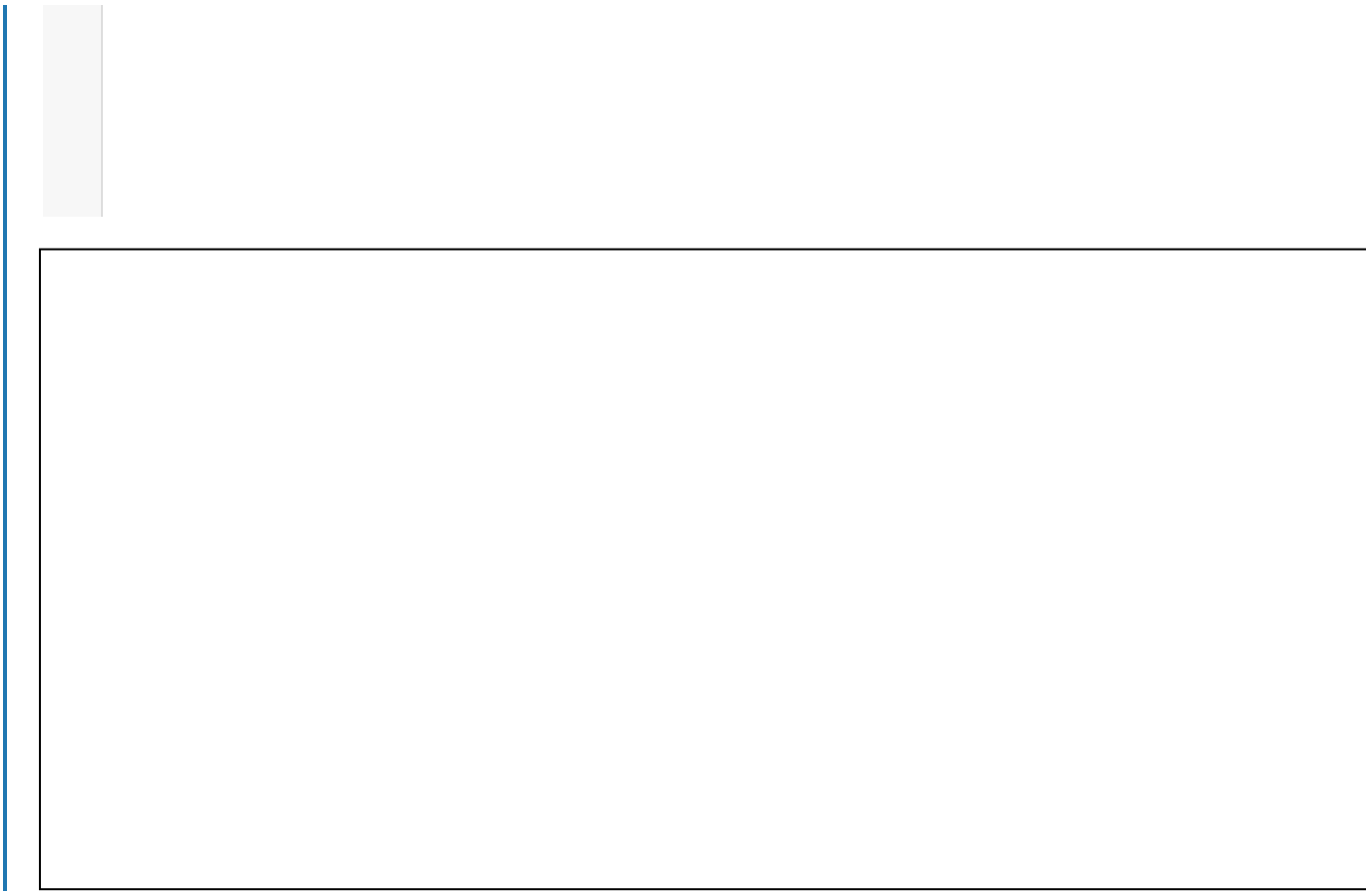
Po dziesięciokrotnym wylosowaniu danej liczby zostaje ona wypisana na standardowym wyjściu.

Twoje zadania

1. Program ma wylosować liczbę 19 i ją wypisać dziesięć razy (od nowej linii).

```cpp
#include <iostream>


int main() {

    // Skorzystaj z poniższej linijki do wypisania prawidłowo wylosowanej liczby za każdym razem:
    // std::cout << liczba << std::endl;


    return 0;
}
```

```
1
```



Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Generowanie liczb pseudolosowych w języku C++

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi, w tym: znajomość zasad działania urządzeń cyfrowych i sieci komputerowych oraz wykonywania obliczeń i programów.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres podstawowy. Uczeń:

1) planuje kolejne kroki rozwiązywania problemu, z uwzględnieniem podstawowych etapów myślenia komputacyjnego (określenie problemu, definicja modeli i pojęć, znalezienie rozwiązania, zaprogramowanie i testowanie rozwiązania).

4) porównuje działanie różnych algorytmów dla wybranego problemu, analizuje algorytmy na podstawie ich gotowych implementacji;

5) sprawdza poprawność działania algorytmów dla przykładowych danych.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

1) w zależności od problemu rozwiązuje go, stosując metodę wstępującą lub zstępującą;

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

Zakres podstawowy. Uczeń:

1) projektuje i programuje rozwiązania problemów z różnych dziedzin, stosuje przy tym: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje z parametrami i bez parametrów, testuje poprawność programów dla różnych danych; w szczególności programuje algorytmy z punktu I.2);

2) do realizacji rozwiązań problemów prawidłowo dobiera środowiska informatyczne, aplikacje oraz zasoby, wykorzystuje również elementy robotyki;

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

1) projektuje i tworzy rozbudowane programy w procesie rozwiązywania problemów, wykorzystuje w programach dobrane do algorytmów struktury danych, w tym struktury dynamiczne i korzysta z dostępnych bibliotek dla tych struktur;

3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Przeanalizujesz, jak generować liczby pseudolosowe w języku C++.
- Napiszesz program w języku C++, który wygeneruje tyle liczb losowych, ile zechce użytkownik, i wypisze je na ekranie.
- Wykonasz kilka ćwiczeń, w których sprawdzisz swoje umiejętności w zakresie generowania liczb pseudolosowych.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda;
- oprogramowanie dla języka C++, w tym kompilator GCC/G++ 4.5 (lub nowszej wersji) i Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Generowanie liczb pseudolosowych w języku C++". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj".
2. Chętny lub wybrany uczeń przygotowuje rozwiązanie polecenia nr 1 z sekcji „Symulacja interaktywna". Będzie pełnił rolę eksperta podczas zajęć.

Faza wstępna:

1. Nauczyciel wyświetla i odczytuje temat lekcji oraz cele zajęć. Prosi uczniów o sformułowanie kryteriów sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Uczniowie tworzą pytania dotyczące tematu zajęć, na które odpowiedzą w trakcie lekcji.

Faza realizacyjna:

1. Odnosząc się do treści zawartej w sekcji „Przeczytaj", uczniowie w parach konstruują alternatywny przykład, definiując samodzielnie problem, rozwiązanie i ewentualną implementację. Rezultaty omawiane są na forum klasy.
2. **Praca z multimedium.** Uczniowie pracują w parach. Wykonują polecenie 1 z sekcji „Symulacja interaktywna". Wyniki swojej pracy porównują z rozwiązaniem

przedstawionym w prezentacji. Następnie uczniowie zapoznają się z poleceniem 3, zapisują problemy i pytania z nim związane. Po czym następuje dyskusja, w trakcie której nauczyciel wyjaśnia niezrozumiałe kroki procedury.
3. **Ćwiczenie umiejętności.** Poszukiwanie najefektywniejszego rozwiązania problemu. Uczniowie wykonują w parach ćwiczenie nr 1-3 z sekcji „Sprawdź się", a następnie porównują swój kod omawiając go wspólnie na forum. Nauczyciel ocenia efektywność zastosowanego rozwiązania.

Faza podsumowująca:

1. Na koniec zajęć nauczyciel raz jeszcze wyświetla na tablicy temat lekcji i cele zawarte w sekcji „Wprowadzenie". W odniesieniu do ich realizacji dokonuje szczegółowej oceny rozwiązania zastosowanego przez wybranego ucznia.
2. Na koniec zajęć z programowania w C++ nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak...".

Praca domowa:

1. Uczniowie wykonują pracę domową z sekcji „Przeczytaj".

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka C++.
- Oficjalna dokumentacja techniczna dla kompilatora GCC/G++ 4.5 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Wskazówki metodyczne:

- Uczniowie mogą wykorzystać treści w sekcjach: „Przeczytaj", „Symulacja interaktywna", „Sprawdź się" jako materiał do lekcji powtórkowej.