Algorytmy i programowanie

Marcin Żurowski

11 marzec 2021

Plan zajęć

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
- Oceny:

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
- Oceny:

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po 0 — 5 punktów (5 punktów po terminie)
- Oceny:

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{5}$ punktu po terminie)
- Oceny:

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - 50 + % punktów - 3
  - 60 + % punktów - 3.5
  - 70 + % punktów - 4
  - 80 + % punktów - 4.5
  - 90 + % punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 + \%$ punktów - 3
  - $60 + \%$ punktów - 3.5
  - $70 + \%$ punktów - 4
  - $80 + \%$ punktów - 4.5
  - $90 + \%$ punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 + \%$ punktów - 3
  - $60 + \%$ punktów - 3.5
  - $70 + \%$ punktów - 4
  - $80 + \%$ punktów - 4.5
  - $90 + \%$ punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 +$ % punktów - 3
  - $60 +$ % punktów - 3.5
  - $70 +$ % punktów - 4
  - $80 +$ % punktów - 4.5
  - $90 +$ % punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 +$ % punktów - 3
  - $60 +$ % punktów - 3.5
  - $70 +$ % punktów - 4
  - $80 +$ % punktów - 4.5
  - $90 +$ % punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 +$ % punktów - 3
  - $60 +$ % punktów - 3.5
  - $70 +$ % punktów - 4
  - $80 +$ % punktów - 4.5
  - $90 +$ % punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 +$ % punktów - 3
  - $60 +$ % punktów - 3.5
  - $70 +$ % punktów - 4
  - $80 +$ % punktów - 4.5
  - $90 +$ % punktów - 5

Zaliczenie

- Obecność:
  - minimalnie 16 obecności
  - 3 zajęcia nieusprawiedliwione
  - tydzień na przyniesienie usprawiedliwienia
- Punkty:
  - 2 kolokwia po 25 punktów (poprawka średnia arytmetyczna)
  - 10 zadań programistycznych po $0 - 3$ punktów ($\frac{1}{2}$ punktu po terminie)
- Oceny:
  - $50 +$ % punktów - 3
  - $60 +$ % punktów - 3.5
  - $70 +$ % punktów - 4
  - $80 +$ % punktów - 4.5
  - $90 +$ % punktów - 5

Literatura

- Jerzy Grębosz, "Symfonia C++ Standard", Wydawnictwo Edition (2000)
- Ewa Palka, "Elementy algorytmiki dla początkujących", Wydawnictwo Naukowe UAM, Poznań (2013)

Literatura

- Jerzy Grębosz, "Symfonia C++ Standard", Wydawnictwo Edition (2000)
- Ewa Palka, "Elementy algorytmiki dla początkujących", Wydawnictwo Naukowe UAM, Poznań (2013)

Problemy

- decyzyjne
- optymalizacyjne
- wyszukiwania

Problemy

- decyzyjne
- optymalizacyjne
- wyszukiwania

Problemy

- decyzyjne
- optymalizacyjne
- wyszukiwania

Problemy

- decyzyjne
- optymalizacyjne
- wyszukiwania

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy instancję problemu

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy **instancję problemu**

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy **instancję problemu**

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy instancję problemu

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy instancję problemu

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy instancję problemu

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy instancję problemu

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy **instancję problemu**

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy **instancję problemu**

Problem definicja

Problem:

- parametry
- pytanie

Przykład:

- parametry: mamy dane ciąg n liczb naturalnych $(n_1, n_2, \ldots, n_n)$
- pytanie:
  - czy istnieje najmniejszy element?
  - znajdź pierwszy najmniejszy element
  - znajdź najmniejszy element

Jeżeli do wszystkich zmiennych wstawimy wartości to uzyskamy **instancję problemu**

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Algorytm

Sekwencja instrukcji elementarnych rozwiązująca dowolną instancję problemu

Własności algorytmu:

- precyzyjny opis
- determinizm (z wyjątkiem algorytmów randomizowanych)
- skończoność
- poprawność
- ogólność

Reprezentacja algorytmów

- język naturalny
- lista kroków
- schemat blokowy
- pseudokod
- język programowania

Reprezentacja algorytmów

- język naturalny
- lista kroków
- schemat blokowy
- pseudokod
- język programowania

Reprezentacja algorytmów

- język naturalny
- lista kroków
- schemat blokowy
- pseudokod
- język programowania

Reprezentacja algorytmów

- język naturalny
- lista kroków
- schemat blokowy
- pseudokod
- język programowania

Reprezentacja algorytmów

- język naturalny
- lista kroków
- schemat blokowy
- pseudokod
- język programowania

Środowisko programistyczne

- Dev-c++ 5.11 IDE (Integrated Development Environment) https://www.instalki.pl/programy/download/Windows/srodowiska_programistyczne/Dev-C_5.html
- Pierwszy program

```
#include <iostream>
int main(int argc, char** argv) {
 std::cout << "czesc" << std::endl;
 return 0;
}
```

Środowisko programistyczne

- Dev-c++ 5.11 IDE (Integrated Development Environment) https://www.instalki.pl/programy/download/Windows/srodowiska_programistyczne/Dev-C_5.html
- Pierwszy program

```
#include <iostream>
int main(int argc, char** argv) {
 std::cout << "czesc" << std::endl;
 return 0;
}
```

Komentarze

- //
- /* */

Komentarze

- //
- /* */

Typy

- `int -23`
- `double -8.0`
- `char`
  `'A' '\b' '\t' '\n' '\r' '\"' '\'' '\\'`
- `bool false true`

Typy

- `int -23`
- `double -8.0`
- `char`
  `'A' '\b' '\t' '\n' '\r' '\"' '\'' '\\'`
- `bool false true`

Typy

- `int -23`
- `double -8.0`
- `char`
  `'A' '\b' '\t' '\n' '\r' '\"' '\'' '\\'`
- `bool false true`

Typy

- `int -23`
- `double -8.0`
- `char`
  `'A' '\b' '\t' '\n' '\r' '\"' '\'' '\\'`
- `bool false true`

Zmienna i stała

- `int a = 12;`
- `const int A = 12;`

Zmienna i stała

- `int a = 12;`
- `const int A = 12;`

Wyjście

```
std::cout << "czesc" << std::endl;
```

Wejście

```
int zmienna;
std::cin >> zmienna;
```

Wejście

```
using namespace std;
int zmienna;
cin >> zmienna;
cout << zmienna << endl;
```

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar sizeof()
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie `(<typ>)`
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie `(<typ>)`
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie `(<typ>)`
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie `(<typ>)`
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

Operatory

- arytmetyczne + - * / %
- jedno argumentow + -
- inkrementacja i dekrementacja ++ --
- przypisanie =
- logiczne < <= > >= == !=
- predykaty && || !
- bitowe << >> & | ^ ~
- przypisanie += -= *= /= %= &= |= ^= <<= >>=
- warunkowy ?:
- rozmiar `sizeof()`
- rzutowanie (<typ>)
- przecinek ,

1

Wyznacz wartości zmiennych `i`, `j`, `k` w następujących przypadkach:

1.
```
i = 10
k = i + 6
i = 2 * i - 5
j = i + k - 3
```
2.
```
i = 10
i = 2 * i - 5
k = i + 6
j = i + k - 3
```
3.
```
k = i + 6
i = 10
i = 2 * i - 5
j = i + k - 3
```

1

Wyznacz wartości zmiennych `i`, `j`, `k` w następujących przypadkach:

1. ```
   i = 10
   k = i + 6
   i = 2 * i - 5
   j = i + k - 3
   ```
2. ```
   i = 10
   i = 2 * i - 5
   k = i + 6
   j = i + k - 3
   ```
3. ```
   k = i + 6
   i = 10
   i = 2 * i - 5
   j = i + k - 3
   ```
   ```

1

Wyznacz wartości zmiennych `i`, `j`, `k` w następujących przypadkach:

1. ```
   i = 10
   k = i + 6
   i = 2 * i - 5
   j = i + k - 3
   ```
2. ```
   i = 10
   i = 2 * i - 5
   k = i + 6
   j = i + k - 3
   ```
3. ```
   k = i + 6
   i = 10
   i = 2 * i - 5
   j = i + k - 3
   ```
   ```

2

Zakładając, że zmiennym `a`, `b`, `c`, `x` zostały przypisane wartości. Zapisz ciąg instrukcji wyliczających wartość wielomianu: $w(x) = ax^2 + bx + c$.

3

Dane są dwie zmienne a i b, którym wcześniej przypisano liczby. Napisz ciąg instrukcji "zamieniający" wartość tych zmiennych. Dokładnie, liczba, która jest przypisana a, ma stać się wartością zmiennej b, a liczba, która jest przypisana zmiennej b, ma stać się wartością zmiennej a.