C++

Tablice

Napisy

Funkcje i procedury

Pliki

Funkcje i procedury

- Poniekąd takie samo jak w Javie:

```
 7  double pot(double podstawa, int wykladnik)
 8  {
 9      double wynik= 1;
10      for(int i = 0; i < wykladnik; ++i) wynik=podstawa*wynik;
11      return wynik;
12  }
13
14
15  int main(int argc, char *argv[])
16  {
17  cout << "2 do potegi 3 wynosi " << pot(2,3)<<endl;
18  system("PAUSE");
19  return 0;
```

Funkcje i procedury

- Poniekąd, bo:
  1. Można zwrócić kilka wartości na raz ☺
  2. Parametry mogą być przekazywane przez **referencję** lub przez **wartość**
  3. Funkcje mogą mieć zmienną liczbę parametrów
  4. Mamy do dyspozycji funkcje **inline**

Referencja vs wartość

```
 7   void wartosc(int co)
 8   {
 9       ++co;
10      return;
11   }
12
13   void referencja(int &co)
14   {
15       ++co;
16      return;
17   }
18
```

```
int main(int argc, char *argv[])
{
    int liczba = 0;
cout << "Na poczatku liczba wynosi: "<< liczba << endl;
wartosc(liczba);
cout << "Ale wywolujemy funkcje wartosc i liczba wynosi juz: " << liczba << endl;
referencja(liczba);
cout << "Lecz dopiero referencja zmieni wartosc na " << liczba <<endl;
system("PAUSE");
return 0;
}
```

Zmienna liczba parametrów

```
 4  #include <cstdarg>
 5  using namespace std;
 6
 7  int mnoz (int pierwszy, ...)
 8   {
 9     va_list arg;
10     int iloczyn = 1, t;
11     va_start (arg, pierwszy);
12     for (t = pierwszy; t; t = va_arg(arg, int)) {
13       iloczyn *= t;
14     }
15     va_end (arg);
16     return iloczyn;
17
18   }
19  int main(int argc, char *argv[])
20  {
21      cout << mnoz(1.0,2,3,4,5,0) <<endl;
22  system("PAUSE");
```

Tablice

- Podobne do Javy:

typ nazwa[rozm]

```
21    typ nazwa_tablicy[rozmiar];
22
23    int tablica[10];
24
25    int tablica[4] = {2,4,6,8};
26
27    int tablica[100] = {1,};
28
29    int tablica[] = {1, 2, 3, 4, 5, 6, 7, 10};
30
```

```
23  int main(int argc, char *argv[])
24  {
25      int tablica[5] = {1,2,3,4,5};
26      for (int i=0; i<(sizeof tablica / sizeof *tablica); ++i) cout << tablica[i] <<endl;
27      system("PAUSE");
28      return 0;
29  }
```

Jeden mały myk

```
 7 void test(int tab[]) {
 8       for(int i=0;i<10;++i) tab[i]+=1;
 9       }
10
11 int main(int argc, char *argv[])
12 {
13 int tab[10]={1,1,1,1,1,1,1,1,1,1};
14 cout << tab[0] << endl;
15 test(tab);
16 cout << tab[0] << endl;
17
18 system("PAUSE");
19 return 0;
```

Drugi mały myk (C++11)

```
int moja_tablica[5] = {1, 2, 3, 4, 5};
for(int &x : moja_tablica)
{
  x *= 2;
}
```

Ćwiczenia

- Zadeklaruj tablicę o rozmiarze 100. Wypełnij tablicę zgodnie z regułami Ciągu Fibbonacciego (pierwszy i drugi element = 1, każdy następny to suma dwóch poprzednich: 1,1,2,3,5,8, itd.).
- Napisz program, który zamienia liczbę dziesiętną podaną przez użytkownika na liczbę binarną i szesnastkową.
- Program „zgadnij moją liczbę”. Program losuje liczbę z zakresu 1...100, a naszym zadaniem jest zgadnąć tą liczbę na podstawie „za dużo”, „za mało”. Po zgadnięciu program wyświetla liczbę prób.
- Wypisz całą tablicę ASCII na ekran (każdy znak w nowej linijce opatrzony „numerkiem”) i zapisz ją do tablicy o nazwie tab_ASCII w programie.

Ćwiczenia

1. Przerób grę w statki tak, aby to komputer losował jednomasztowce. Nie pokazuj użytkownikowi planszy, niech strzela i spróbuje trafić. Na końcu – wyświetl mu ile strzałów potrzebował.

- Napisać program, który:
  - utworzy tablicę 20 liczb całkowitych i wypełni ją wartościami losowymi z przedziału [–10, . . . , 10],
  - wypisze na ekranie zawartość tablicy,
  - wyznaczy najmniejszy oraz największy element w tablicy,
  - wyznaczy średnią arytmetyczną elementów tablicy,
  - wyznaczy ile elementów jest mniejszych, ile większych od średniej.
  - wypisze na ekranie zawartość tablicy w odwrotnej kolejności, tj. od ostatniego do pierwszego.

1. Napisać program do sumowania macierzy: deklarujemy dwie tablice (tab1, tab2) o rozmiarze 7x7, wypełniamy je losowymi liczbami, a następnie do tab3 wpisujemy sumę elementów na odpowiednich pozycjach w tab1 i tab2.

- Napisz program, który pobiera od użytkownika dodatnią liczbę naturalną n i tworzy tablicę a zmiennych typu logicznego (boolean) o rozmiarze n×n.
- Następnie program powinien wypełnić utworzoną tablicę, tak by a[i][j] = true jeżeli liczby (i+1) oraz (j+1) są względnie pierwsze, tzn. nie mają wspólnych dzielników poza 1.
- Tak utworzoną tablicę należy wypisać na ekranie, przy czym dla wartości true należy wyświetlić znak "+", natomiast dla wartości false znak ".". Przykład:

```
Podaj liczbę (> 0): 10
    1  2  3  4  5  6  7  8  9 10
 1  +  +  +  +  +  +  +  +  +  +
 2  +  .  +  .  +  .  +  .  +  .
 3  +  +  .  +  +  .  +  +  .  +
 4  +  .  +  .  +  .  +  .  +  .
 5  +  +  +  +  .  +  +  +  +  .
 6  +  .  .  .  +  .  +  .  .  .
 7  +  +  +  +  +  +  .  +  +  +
 8  +  .  +  .  +  .  +  .  +  .
 9  +  +  .  +  +  .  +  +  .  +
10  +  .  +  .  .  .  +  .  +  .
```

Ćwiczenia

- Idą święta – narysuj choinkę z gwiazdek (rozmiar podaje użytkownik):

- Napisać program, który pobiera od użytkownika liczbę całkowitą dodatnią, a następnie wyświetla na ekranie kolejno wszystkie liczby nieparzyste nie większe od podanej liczby. Przykład, dla 15 program powinien wyświetlić 1, 3, 5, 7, 9, 11, 13, 15.
- Napisać program, który pobiera od użytkownika ciąg liczb całkowitych. Pobieranie danych kończone jest podaniem wartości 0 (nie wliczana do danych). W następnej kolejności program powinien wyświetlić sumę największej oraz najmniejszej z podanych liczb oraz ich średnią arytmetyczną.
  Przykład: Użytkownik podał ciąg: 1, -4, 2, 17, 0.
  Wynik programu:
  13 // suma min. i maks.
  3.2 // średnia

12. Napisać program działający w trybie konsolowym (tekstowym) i rysujący na ekranie prostokąt. Użytkownik podaje znak wypełnienia prostokąta, pozycję lewego górnego rogu prostokąta (x, y) oraz długości boków prostokąta (ab). Przyjmujemy, że lewy górny n konsoli ma współrzędne (x, y) = (1, 1).
Przykład: x=6, y=3, a=4, b=6, zn='x'

```
>
>
> _____xxxxxx
> _____xxxxxx
> _____xxxxxx
> _____xxxxxx

ozn.
> - nowa linia,
_ - znak spacji.
```

12. Napisać program, dla podanej liczby całkowitej wyświetla jej dzielniki. Przykładowo, dla liczby 21 dzielniki to: 1, 3, 7, 21.
13. Napisać program, który sprawdza, czy podana liczba całkowita n, n > 1, jest liczbą pierwszą.

Napisy

Tu będzie trudniej niż w Javie!

Napisy

Tu będzie trudniej niż w Javie!

Żartowałem ☺

Napisy

Tu będzie trudniej niż w Javie!

Żartowałem ☺

Ale nie tak do końca. Napisy w C są traktowane jako tzw. „null terminated char array”.
Gorąco polecam: http://pl.wikibooks.org/wiki/C/Napisy
Aczkolwiek w ramach upraszczania sobie życia, przejdziemy od razu do napisów w C++.

Napisy: 0-terminated

- Napisy traktowane jako tablice zakończone znakiem o kodzie 0:

```
 1  #include <cstdlib>
 2  #include <iostream>
 3  #include <cstring>
 4
 5  using namespace std;
 6
 7  int main(int argc, char** argv) {
 8
 9  char tekst[90] = "Ten tekst musi byc krotszy niz 89 znakow";
10  char nap[10] = "123456789";
11  char napis[] = "Ala ma kota";
12
13  cout << tekst << endl << napis <<endl << nap << endl;
14
```

Napisy: 0-terminated

- Porównywanie: strcmp(str1, str2): zwraca -, + lub 0
- Kopiowanie: strcpy(str1, str2) – znaczy to samo co intuicyjne: str1 = str2;
- Konkatenacja: strcat(str1, str2) – znaczy to samo co intuicyjne: str1 = str1 + str2;
- tolower, toupper, islower, isupper – trywialne
- Konwersje:
  - atol, strtol - zamienia łańcuch na liczbę całkowitą typu long
  - atoi - zamienia łańcuch na liczbę całkowitą typu int
  - atoll, strtoll - zamienia łańcuch na liczbę całkowitą typu long long (64 bity); dodatkowo istnieje przestarzała funkcja atoq będąca rozszerzeniem GNU,
  - atof, strtod - przekształca łańcuch na liczbę typu double

Ćwiczenia

1. Zadeklaruj napis c-string o długości 10 i przypisz mu 11 znaków (najpierw w deklaracji, a potem za pomocą konkatenacji)
2. Zmień małe na wielkie litery i odwrotnie w napisie przeczytanym z klawiatury.
3. Napisz funkcję int[] HtmlToRGB(char[] kodKoloru)
4. Napisz funkcję int IsPali(char[] palindrom)

Napisy: the easy way

- Jesteśmy uratowani! W C++ jest string!

```
 9 string s;
10 cin >> s; //sproboj tu podac cos ze spacja :)
11 cin.ignore();
12 fflush(stdin);
13 cout << s << endl;
14 getline(cin,s);
15 cout << s << endl;
16 string drugi="Ala ma kota";
17 if(s==drugi) cout << "Yupi!"; else cout << "Niestety";
18 cout << endl;
19 s+=" i psa";
20 cout << s << endl;
```

```
35
36 string zdanie="Byc, albo nie byc - oto jest pytanie";
37 cout << zdanie <<endl;
38 for(int i=0;i<zdanie.size();i+=2) { //pokemonizator
39         zdanie.at(i) = toupper(zdanie.at(i));
40         }
41 cout << zdanie;
```

```
24 string csv;
25 do {
26      cin >> csv;
27      cout << csv << endl;
28 }
29 while(csv!=";");
30
31 csv="";
32 getline(cin,csv,'$');
33 cout << csv << endl;
```

Napisy: the easy way

- Pozostałe metody:
  - compare(str1, str2) – analogiczne do tego z C
  - c_str(str) – zwraca „stary” łańcuch oterminated
  - find, find_first_not_of, find_first_of, …
  - str1.replace(start,length,sourceForReplacement)
  - reverse
  - substr
  - swap

Ćwiczenia

1. Czym różni się operator [] od at? Sprawdź to praktycznie ☺
2. Zaimplementuj proste szyfry harcerskie: GA-DE-RY-PO-LUKI, PO-LI-TY-KA-RE-NU, KA-CE-MI-NU-TO-WY. Wyświetla się menu, gdzie użytkownik wybiera szyfr. Następnie komunikat „Podaj ciąg wejściowy”, który jest odczytywany przez program a następnie zmieniane litery zgodnie ze wzorcem szyfru.
3. Przeczytaj od użytkownika ciąg tekstu zakończony znakiem kropki (.) Następnie podziel tenże tekst na słowa i wszystkie dłuższe od 4 znaków zapisz do oddzielnej tablicy stringów. Na końcu wypisz na ekran napis złożony z 2 i 3 znaku każdego elementu w tej nowej tablicy.
4. Napisz funkcję sprawdzającą czy wyraz jest palindromem. Tu konkurs na najkrótszy program ☺

Ćwiczenia

- Polecenie: napisz funkcję, który generuje informacje o porach, w których powinien rozlegać się dźwięk dzwonka.
- Po pierwsze, godzina rozpoczęcia pierwszej lekcji może być zmienna i niekoniecznie musi być nią 8:00. Wynika z tego, że informację tę program musi przyjąć w postaci pierwszej danej wejściowej. Również długości przerw są zmienne, co akurat jest zupełnie zrozumiałe. Jedno tylko pozostaje niezmienne – długość lekcji, która zawsze trwa równo 45 minut.
- Dane wejściowe: string określający godzinę rozpoczęcia zajęć, tablica intów zawierająca przerwy, wyrażonych w minutach (całkowite t: t >0 i t<=10080)
- Dane wyjściowe: jeden wiersz tekstu, a w nim lista rozdzielonych przecinkami pór uruchomienia dzwonka w formacie HH:MM
- Przykład:
  - Wejście:
  - Tablica przerwy[] = {15, 15, 15}
  - cout << dzwonki(„08:00”, przerwy);
- Wyjście:
  - 08:00,08:45,09:00,09:45,10:00,10:45,11:00,11:45

Obsługa plików

- I znowu – można jak w C, za pomocą „starych” struktur i metod:

```c
#include <stdio.h>
#include <stdlib.h>

int main ()
{
  FILE *fp; /* używamy metody wysokopoziomowej - musimy mieć zatem identyfikator pliku, uwaga na gwiazdkę! */
  char tekst[] = "Hello world";
  if ((fp=fopen("test.txt", "w"))==NULL) {
    printf ("Nie mogę otworzyć pliku test.txt do zapisu!\n");
    exit(1);
    }
  fprintf (fp, "%s", tekst); /* zapisz nasz łańcuch w pliku */
  fclose (fp); /* zamknij plik */
  return 0;
}
```

… ale my się uczymy C++ i mamy strumienie:

Zapis(ofstream)

```
4  int main()
5  {
6      char znak;
7      std::ofstream plik("testowy.txt");
8      do {
9          znak = std::cin.get();
10         plik << znak;
11     } while (znak != '.');
12     return 0;
13 }
```

Odczyt(ifstream)

```
1  #include <iostream>
2  #include <fstream>
3
4  int main()
5  {
6      char znak;
7      std::ifstream plik("testowy.txt");
8      do {
9          znak = plik.get();
10         std::cout << znak;
11     } while (znak != '.');
12
13 system("PAUSE");
14 return 0;
15 }
```

```
6  std::string linia;
7  std::ifstream plik("testowy.txt");
8
9      do {
10         getline(plik,linia);
11         std::cout << linia <<std::endl;
12     } while (!plik.eof());
```

Pliki i strumienie

```
1  #include <iostream>
2  #include <fstream>
3
4  int main()
5  {
6  std::string linia;
7  std::ofstream plik;
8  plik.open("testowy.txt",std::ofstream::app);
9  for(int i=0;i<10;++i) {
10 plik << rand()%10 <<std::endl;
11 }
12 std::ifstream plikO;
13 plikO.open("testowy.txt");
14    do {
15       getline(plikO,linia);
16       std::cout << linia <<std::endl;
17    } while (!plikO.eof());
18
19 system("PAUSE");
20 return 0;
21 }
```

http://www.cplusplus.com/reference/iostream/ifstream/

http://www.cplusplus.com/reference/iostream/ofstream/

Odczyt

- **int get();** - zwraca znak ze strumienia (lub EOF jeśli skończył się plik)
- **istream& get(char& c);**
  - wpisuje odczytany ze strumienia znak (koniec pliku należy sprawdzić przez eof()).
- **istream& get(char* buf, int len, char eot = '\n');** - pobiera ze strumienia do `buf' maksymalnie `len' znaków i aż do napotkania znaku `eot' (tablica znaków jest terminowana zerem)
- **istream& getline(char* buf, int len, char eot = '\n');** - jak powyższa get, ale znak `eot' nie jest zapisywany w `buf'
- **istream& read(char* ptr, long len);**
- **istream& read(void* ptr, long len);**
  - odczytują ze strumienia maksymalnie `len' znaków do `ptr'; jest to jak widać funkcja do odczytu danych binarnych, zatem tablica NIE jest terminowana zerem!
- **int peek();** - podgląda następny znak w strumieniu (strumień nie ulega zmianie)
- **istream& unget();**- wstawia znak z powrotem do strumienia ☺

Biblioteka standardowa

- assert.h - makra do asercji
- ctype.h - klasyfikacje znaków typu char (isspace, isalpha itd.)
- errno.h - deklaracja errno
- limits.h - makra określające granice dla typów ścisłych
- locale.h - definicje lokali
- math.h - funkcje matematyczne
- stdarg.h - narzędzia dla funkcji o zmiennej liście argumentów
- stddef.h - standardowe definicje (ptrdiff_t i size_t głównie)
- stdio.h - operacje wejścia/wyjścia
- stdlib.h - zespół funkcji użytkowych string.h - funkcje operujące na tablicach znaków
- time.h - narzędzia do odczytywania, interpretacji i prezentacji czasu

Zadanie domowe lub ćwiczenia

1. Poczytaj o instrukcja preprocesora (#define, #include, #error, makra). Za pomocą tej wiedzy napisz kilka szybkich makr automatyzujących liczenie np. sinusa lub cosinusa.
2. Zaimplementuj Szyfr Cezara: kodowanie i dekodowanie (http://pl.wikipedia.org/wiki/Szyfr_Cezara) wraz z obsługą plików (szyfrogram, deszyfrogram). Wymyśl sobie własną technikę zapisu.
3. Odczytaj tekst wpisywany przez użytkownika a następnie wypisz statystykę analizy częstości występowania poszczególnych liter (od ‘a’ do ‘z’) zamieniając wielkie litery na małe.
4. Wypełnij tablicę 10 x 10 tabliczką mnożenia… ale w kodzie trójkowym.
5. Napisz program odwrotny do „zgadnij moją liczbę”: człowiek wymyśla liczbę, program ma ją znaleźć.

Zadanie domowe

- Polecenie: napisz program, który skróci dowolną nazwę zmiennej do maksymalnej długości n w sposób opisany poniższym algorytmem. Napisz nazwę zmiennej w postaci, w której życzyłbyś sobie ją widzieć – używaj tylko liter, cyfr oraz znaków '_' (podkreślenie) i '$' (dolar);
- jeśli zmienna zawiera znaki niedopuszczalne – zwróć w wyniku „o”;
- jeśli długość nazwy jest mniejsza równa n, możesz jej użyć i nic musisz robić nic więcej
- w przeciwnym wypadku usuwaj z nazwy, począwszy od końca, wszystkie znaki, które nie są literami i cyframi – w chwili, w której długość nazwy osiągnie n, możesz zakończyć pracę i użyć nazwy zmiennej
- jeśli długość nazwy nadal jest większa od n, usuwaj z niej, począwszy od końca, kolejne cyfry - w chwili, w której długość nazwy osiągnie n możesz zakończyć pracę i użyć nazwy zmiennej
- jeśli długość nazwy nadal jest większa od n, usuwaj z niej, począwszy od początku, kolejne samogłoski z wyjątkiem pierwszej (chodzi o to, by w nazwie została chociaż jedna samogłoska, o ile jakakolwiek została użyta) - w chwili, w której długość nazwy osiągnie n, możesz zakończyć pracę i użyć nazwy zmiennej
- jeśli długość nazwy nadal jest większa od n, usuwaj z niej znaki od końca, począwszy od przedostatniego - w chwili, w której długość nazwy osiągnie n, możesz zakończyć pracę i użyć nazwy zmiennej
- Dane wejściowe: plik tekstowy zawierający w kolejnych porcjach danych:
  2 wiersze, zawierające kolejno:
  maksymalną dopuszczalną długość zmiennej (n: n >= 1 i n <= 65535)
  nazwę zmiennej (o długości l), która będzie podlegać skracaniu (l: l >= 1 i l <= 65636)
- Dane wyjściowe:
- Po jednym wierszu na każdy przypadek testowy, zawierający nazwę zmiennej poddaną algorytmowi skracania.
- Przykład:
- Wejście:
  - 7
  - ALA_MA_KOTA_I_2_PSY
  - 10
  - SUPER_ZMIENNA(
- Wyjście:
  - ALMKTPS
  - o