Informatyka

4.Wstęp do programowania w C/C++

Opracował: Maciej Penar

Spis treści

[1. Wprowadzenie do C 3](#_Toc496799997)

[2. Obok C 5](#_Toc496799998)

# Wprowadzenie do C

Napisać w **C** aplikacje konsolowe:

1. Zapytaj użytkownika o imię, nazwisko i wiek. Wyświetlić „Witaj [imię] [nazwisko], masz [wiek] lat”.
   * Zwróć uwagę na walidację wejścia: wiek musi mieć sens, imię/nazwisko nie mogą być puste.
   * *Dla chętnych,* ***można skorzystać z C++****: rozszerzyć walidację w zadaniu 1 w oparciu o wyrażenia regularne tak żeby imię i nazwisko zaczynało się od dużej litery i składało się tylko z liter.*

|  |
| --- |
| Podpowiedź do 1b |
| Wyrażenia regularne, ściągawka: https://regexone.com/  Dokumentacja link: <http://en.cppreference.com/w/cpp/regex>)  W sumie trzy linijki:  Pierwsza: #include <regex>  Druga: std::basic\_regex<char> reg(CAŁA ZAGADKA, std::regex::icase);  Trzecia: if (std::regex\_match<char>(text, reg)) |

1. Brytyjczycy są dziwni i podają wagę w kamieniach. 1 kamień = 6.35029318 kilograma. Napisz program który pyta użytkownika o jednostkę w jakiej poda wagę oraz wartość wagi. Wypisz przeliczenie na kilogramy/kamienie. Użyj stałych symbolicznych. Napisz dwie wersje programu (lub jedną sprytną w której możliwa jest łatwa podmiana typu) – wykorzystującą double i float. Czy są różnice.
2. Napisz funkcję wykonującą potęgowanie liczby 2 (wykładnik to parametr):
   * Przez pętle
   * Przez przesunięcie bitowe
   * *Dla chętnych: dodać implementację przez pseudo-przesłonięcie funkcji pow() z <math.h>. Wykonać test która z trzech metod jest szybsza.*

|  |
| --- |
| Podpowiedź do 3c |
| Dołączyć <time.h>.  Poniższy kod wylicza różnicę czasu:  time\_t start,end;  double dif;  time (&start);  // Do some calculation.  time (&end);  dif = difftime (end,start);  Link: https://stackoverflow.com/questions/3557221/how-do-i-measure-time-in-c |

1. Napisz funkcję która oblicza silnie: n!
   * Napisz wersję iteracyjną (wykorzystującą pętle)
   * Napisz wersję rekurencyjną (funkcja wywołująca samą siebie). Spróbuj wykorzystać *„operator trójargumentowy”*
   * Wyrób opinię na temat obu wersji (długość kodu / czytelność kodu / wydajność)
   * Dla jakiego x następują anomalie na typach int, long, long long

|  |
| --- |
| Podpowiedź do 4/4b |
| Operator trójargumentowy ma następującą składnie:  [warunek] ? [wartość gdy prawda] : [wartość gdy fałsz]  Np. zapis **x > 0 ? x : -x** oblicza wartość bezwzględną liczby x  Wzór na silnie: |

1. Napisać strukturę *student* która przechowuje następujące dane: identyfikator, imię, nazwisko, wiek, rok studiów, semestr. Dobrać typy danych oraz uzasadnić ich wybór.
2. Kontynuacja (5): Okazuje się że student może być identyfikowany albo za pomocą:
   * Liczby nieujemnej całkowitej
   * 10 znaków

Zaproponować unię (link: <https://www.tutorialspoint.com/cprogramming/c_unions.htm>) reprezentującą identyfikator. Zainicjalizować unię znakami, sprawdzić:

1. Adres poszczególnych pól
2. Szerokość unii
3. Wartość znaków
4. Wartość liczbową
5. Dany jest plik **iris.csv** (dość popularny, jest tu np. https://github.com/mpenarprz/InformatykaA1/blob/master/dane/iris.csv). Pobrać. Przekierować plik jako dane wejściowe do programu. Zaproponować **strukturę** do przechowywania danych. Napisać program który wyliczy średnią wartość pól „sepal\_length”, „sepal\_width”, „petal\_width”, „petal\_length” w każdej z grup „species”. Wypisać te wartości. Zastanowić się czy trzeba trzymać wszystkie obiekty w pamięci. Wyrzucić nagłówek pliku .csv (można ręcznie).

|  |
| --- |
| Podpowiedź do 7 |
| W Visual Studio by przekierować plik jako wejście wczytywane z scanf() / cin należy:   1. Skopiować plik do folderu w którym jest projekt 2. Wejść we właściwości projektu (PPM->Właściwości) 3. Zakładka Debugowanie 4. W polu „Argumenty polecenia wpisać **< iris.csv** |

1. Kontynuacja 7: jeśli poprzednie zadanie wykonano z wyliczaniem „w locie” – przerobić program tak by buforował rekordy.

# Obok C

|  |
| --- |
| Podpowiedź do kolejnych zadań (1,2) |
| W Visual Studio można zmusić pre-processor by zapisał wynik do pliku, by to wykonać należy:   1. Wejść w ustawienia projektu (alt + enter) 2. Właściwości konfiguracji -> C/C++ -> Preprocessor 3. Zaznaczyć **Przetwarzaj Wstępnie do Pliku TAK** |

Odpowiedz na pytania:

1. Co robi dyrektywa #include
2. Co robi dyrektywa #define:

|  |
| --- |
| #define MAX(a,b) (a>b ? a:b)  void f(){  if (MAX(-1, 1) > 0) {…}  } |

1. Jaki będzie efekt działania kodu:

|  |
| --- |
| int i = 0;  printf("%d", i);  printf("%d", i++);  printf("%d", ++i);  int s = sizeof(i++);  printf("%d", i); |

1. Przedstaw program któremu kończy się pamięć:
   1. Na stercie (Heap)
   2. Na stosie (Stack)
2. Oceń kod:

|  |
| --- |
| #include <iostream>  template  <typename T>  T pomnoz(T a, T b) {  auto output = 0;  auto lim = b >= 0 ? b : -b;  for (int i = 0; i < lim; ++i)  output += a;  return b < 0 ? -output : output;  }    int main()  {  int zmienna;  for (int i = -10; i <= 10; ++i) {  for (int j = -10; j <= 10; ++j) {  zmienna = pomnoz<int>(i, j);  printf("<%d,%d>: %d\n", i, j, zmienna);  }  std::cout<< std::endl;  }  } |

1. Na czym polega przepełnienie liczb całkowitych (Integer Overflow). Napisać program który ilustruje problem. Wykorzystać liczby całkowite ze znakiem oraz bez znaku.
2. Wybrać typ zmiennej oraz zadeklarować następujące dane:
   1. Wiek
   2. **Stałą** PI
   3. Wskaźnik na **stałą** PI
   4. **Stały** wskaźnik na **stałą** PI