

# Spis treści

(Wszystkich trzech tomów)

## Tom I

|                                                            |           |
|------------------------------------------------------------|-----------|
| <b>O Proszę nie czytać tego ! .....</b>                    | <b>1</b>  |
| <b>1 Startujemy ! .....</b>                                | <b>6</b>  |
| 1.1 Pierwszy program .....                                 | 6         |
| 1.2 Drugi program .....                                    | 10        |
| <b>2 Instrukcje sterujące .....</b>                        | <b>15</b> |
| 2.1 Prawda – Fałsz .....                                   | 15        |
| 2.2 Instrukcja warunkowa if .....                          | 16        |
| 2.3 Instrukcja while .....                                 | 18        |
| 2.4 Pętla do...while... .....                              | 19        |
| 2.5 Pętla for .....                                        | 20        |
| 2.6 Instrukcja switch .....                                | 22        |
| 2.7 Instrukcja break .....                                 | 24        |
| 2.8 Instrukcja goto .....                                  | 25        |
| 2.9 Instrukcja continue .....                              | 27        |
| 2.10 Klamry w instrukcjach sterujących .....               | 28        |
| <b>3 Typy .....</b>                                        | <b>30</b> |
| 3.1 Deklaracje typów .....                                 | 30        |
| 3.2 Systematyka typów z języka C++ .....                   | 32        |
| 3.3 Typy fundamentalne .....                               | 32        |
| 3.3.1 Definiowanie obiektów „w biegu” .....                | 34        |
| 3.4 Stałe dosłowne .....                                   | 35        |
| 3.4.1 Stałe będące liczbami całkowitymi .....              | 35        |
| 3.4.2 Stałe reprezentujące liczby zmiennoprzecinkowe ..... | 36        |
| 3.4.3 Stałe znakowe .....                                  | 37        |
| 3.4.4 Stałe tekstowe, albo po prostu stringi .....         | 39        |
| 3.5 Typy pochodne .....                                    | 40        |
| 3.5.1 Typ void .....                                       | 42        |

|       |                                                                 |    |
|-------|-----------------------------------------------------------------|----|
| 3.6   | Zakres ważności nazwy obiektu, a czas życia obiektu .....       | 42 |
| 3.6.1 | Zakres: lokalny .....                                           | 42 |
| 3.6.2 | Zakres: blok funkcji .....                                      | 43 |
| 3.6.3 | Zakres: obszar pliku .....                                      | 43 |
| 3.6.4 | Zakres: obszar klasy .....                                      | 44 |
| 3.7   | Zasłanianie nazw .....                                          | 44 |
| 3.8   | Modyfikator <code>const</code> .....                            | 45 |
| 3.8.1 | Pojedynek: <code>const</code> contra <code>#define</code> ..... | 47 |
| 3.9   | Obiekty <code>register</code> .....                             | 48 |
| 3.10  | Modyfikator <code>volatile</code> .....                         | 49 |
| 3.11  | Instrukcja <code>typedef</code> .....                           | 50 |
| 3.12  | Typy wyliczeniowe <code>enum</code> .....                       | 52 |

## 4 Operatory .....

|       |                                                                                              |    |
|-------|----------------------------------------------------------------------------------------------|----|
| 4.1   | Operatory arytmetyczne .....                                                                 | 54 |
| 4.1.1 | Operator <code>%</code> czyli modulo .....                                                   | 55 |
| 4.1.2 | Jednoargumentowe operatory <code>+</code> i <code>-</code> .....                             | 57 |
| 4.1.3 | Operatory inkrementacji i dekrementacji .....                                                | 57 |
| 4.1.4 | Operator przypisania <code>=</code> .....                                                    | 59 |
| 4.2   | Operatory logiczne .....                                                                     | 60 |
| 4.2.1 | Operatory relacji .....                                                                      | 60 |
| 4.2.2 | Operatory sumy logicznej <code>  </code> i iloczynu logicznego <code>&amp;&amp;</code> ..... | 61 |
| 4.2.3 | Operator negacji: <code>!</code> .....                                                       | 63 |
| 4.3   | Operatory bitowe .....                                                                       | 63 |
| 4.3.1 | Przesunięcie w lewo <code>&lt;&lt;</code> .....                                              | 64 |
| 4.3.2 | Przesunięcie w prawo <code>&gt;&gt;</code> .....                                             | 65 |
| 4.3.3 | Bitowe operatory sumy, iloczynu, negacji, różnicy symetrycznej .....                         | 66 |
| 4.4   | Różnica między operatorami logicznymi, a operatorami bitowymi .....                          | 66 |
| 4.5   | Pozostałe operatory przypisania .....                                                        | 67 |
| 4.6   | Wyrażenie warunkowe .....                                                                    | 68 |
| 4.7   | Operator <code>sizeof</code> .....                                                           | 69 |
| 4.8   | Operator rzutowania .....                                                                    | 70 |
| 4.9   | Operator przecinek .....                                                                     | 71 |
| 4.10  | Priorytety operatorów .....                                                                  | 71 |
| 4.11  | Łączność operatorów .....                                                                    | 74 |

## 5 Funkcje .....

|        |                                                                             |    |
|--------|-----------------------------------------------------------------------------|----|
| 5.1    | Zwracanie rezultatu przez funkcję .....                                     | 78 |
| 5.2    | Stos .....                                                                  | 80 |
| 5.3    | Przesyłanie argumentów do funkcji przez wartość .....                       | 81 |
| 5.4    | Przesyłanie argumentów przez referencję .....                               | 83 |
| 5.5    | Kiedy deklaracja funkcji nie jest konieczna .....                           | 86 |
| 5.6    | Argumenty domniemane .....                                                  | 87 |
| 5.7    | Nienazwany argument .....                                                   | 90 |
| 5.8    | Funkcje <code>inline</code> (w linii) .....                                 | 91 |
| 5.9    | Przypomnienie o zakresie ważności nazw deklarowanych wewnątrz funkcji ..... | 95 |
| 5.10   | Wybór zakresu ważności nazwy i czasu życia obiektu .....                    | 96 |
| 5.10.1 | Obiekty globalne .....                                                      | 96 |



|        |                                                          |     |
|--------|----------------------------------------------------------|-----|
| 5.10.2 | Obiekty automatyczne .....                               | 97  |
| 5.10.3 | Obiekty lokalne statyczne .....                          | 98  |
| 5.11   | Funkcje w programie składającym się z kilku plików ..... | 102 |
| 5.11.1 | Nazwy statyczne globalne .....                           | 106 |
| 5.12   | Funkcje biblioteczne .....                               | 107 |

## **6 Preprocesor .....** **110**

|      |                                                                           |     |
|------|---------------------------------------------------------------------------|-----|
| 6.1  | Na pomoc rodakom.....                                                     | 110 |
| 6.2  | Dyrektywa #define .....                                                   | 112 |
| 6.3  | Dyrektywa #undef.....                                                     | 115 |
| 6.4  | Makrodefinicje .....                                                      | 115 |
| 6.5  | Dyrektywy kompilacji warunkowej .....                                     | 118 |
| 6.6  | Dyrektywa #error.....                                                     | 121 |
| 6.7  | Dyrektywa #line.....                                                      | 122 |
| 6.8  | Wstawianie treści innych plików w tekst kompilowanego właśnie pliku ..... | 123 |
| 6.9  | Sklejacz czyli operator ## .....                                          | 124 |
| 6.10 | Dyrektywa pusta .....                                                     | 125 |
| 6.11 | Dyrektywy zależne od implementacji .....                                  | 125 |
| 6.12 | Nazwy predefiniowane .....                                                | 125 |

## **7 Tablice .....** **128**

|       |                                                      |     |
|-------|------------------------------------------------------|-----|
| 7.1   | Elementy tablicy .....                               | 129 |
| 7.2   | Inicjalizacja tablic .....                           | 131 |
| 7.3   | Przekazywanie tablicy do funkcji .....               | 132 |
| 7.4   | Tablice znakowe .....                                | 136 |
| 7.5   | Tablice wielowymiarowe .....                         | 144 |
| 7.5.1 | Przesyłanie tablic wielowymiarowych do funkcji ..... | 147 |

## **8 Wskaźniki.....** **149**

|        |                                                                           |     |
|--------|---------------------------------------------------------------------------|-----|
| 8.1    | Wskaźniki mogą bardzo ułatwić życie .....                                 | 149 |
| 8.2    | Definiowanie wskaźników .....                                             | 151 |
| 8.3    | Praca ze wskaźnikiem .....                                                | 152 |
| 8.4    | L-wartość.....                                                            | 155 |
| 8.5    | Wskaźniki typu void .....                                                 | 156 |
| 8.6    | Cztery domeny zastosowania wskaźników .....                               | 159 |
| 8.7    | Zastosowanie wskaźników wobec tablic .....                                | 159 |
| 8.7.1  | Ćwiczenia z mechaniki ruchu wskaźnika .....                               | 159 |
| 8.7.2  | Użycie wskaźnika w pracy z tablicą.....                                   | 163 |
| 8.7.3  | Arytmetyka wskaźników .....                                               | 167 |
| 8.7.4  | Porównywanie wskaźników .....                                             | 169 |
| 8.8    | Zastosowanie wskaźników w argumentach funkcji .....                       | 172 |
| 8.8.1  | Jeszcze raz o przesyłaniu tablic do funkcji .....                         | 175 |
| 8.8.2  | Odbieranie tablicy jako wskaźnika.....                                    | 176 |
| 8.8.3  | Argument formalny będący wskaźnikiem do obiektu const .....               | 178 |
| 8.9    | Zastosowanie wskaźników przy dostępie do konkretnych komórek pamięci .... | 181 |
| 8.10   | Rezerwacja obszarów pamięci .....                                         | 181 |
| 8.10.1 | Operatory new i delete albo Oratorium Stworzenie Świata. ....             | 182 |
| 8.10.2 | Dynamiczna alokacja tablicy .....                                         | 186 |
| 8.10.3 | Zapas pamięci to nie jest studnia bez dna .....                           | 189 |
| 8.10.4 | Porównanie starych i nowych sposobów .....                                | 191 |

|        |                                                         |     |
|--------|---------------------------------------------------------|-----|
| 8.11   | Stałe wskaźniki .....                                   | 192 |
| 8.12   | Stałe wskaźniki, a wskaźniki do stałych .....           | 193 |
| 8.13   | Strzał na oślep – Wskaźnik zawsze pokazuje na coś ..... | 194 |
| 8.14   | Sposoby ustawiania wskaźników .....                     | 196 |
| 8.15   | Tablice wskaźników .....                                | 197 |
| 8.16   | Wariacje na temat stringów .....                        | 199 |
| 8.17   | Wskaźniki do funkcji .....                              | 206 |
| 8.17.1 | Ćwiczenia z definiowania wskaźników do funkcji .....    | 209 |
| 8.17.2 | Wskaźnik do funkcji jako argument innej funkcji .....   | 216 |
| 8.17.3 | Tablica wskaźników do funkcji .....                     | 220 |
| 8.18   | Argumenty z linii wywołania programu .....              | 223 |

## **9 Przeladowanie nazw funkcji ..... 227**

|       |                                                                                     |     |
|-------|-------------------------------------------------------------------------------------|-----|
| 9.1   | Co to znaczy: przeladowanie .....                                                   | 227 |
| 9.2   | Bliższe szczegóły przeladowania .....                                               | 231 |
| 9.3   | Czy przeladowanie nazw funkcji jest techniką obiektowo orientowaną? .....           | 233 |
| 9.4   | Linkowanie z modułami z innych języków .....                                        | 235 |
| 9.5   | Przeladowanie a zakres ważności deklaracji funkcji .....                            | 236 |
| 9.6   | Rozważania o identyczności lub odmienności typów argumentów .....                   | 238 |
| 9.6.1 | Przeladowanie a typedef i enum .....                                                | 238 |
| 9.6.2 | Tablica a wskaźnik .....                                                            | 239 |
| 9.6.3 | Pewne szczegóły o tablicach wielowymiarowych .....                                  | 240 |
| 9.6.4 | Przeladowanie a referencja .....                                                    | 242 |
| 9.6.5 | Identyczność typów: T, const T, volatile T.....                                     | 243 |
| 9.6.6 | Przeladowanie a typy: T*, volatile T*, const T* .....                               | 244 |
| 9.6.7 | Przeladowanie a typy: T&, volatile T&, const T& .....                               | 245 |
| 9.7   | Adres funkcji przeladowanej .....                                                   | 246 |
| 9.7.1 | Zwrot rezultatu będącego adresem funkcji przeladowanej.....                         | 248 |
| 9.8   | Kulisy dopasowywania argumentów do funkcji przeladowanych .....                     | 250 |
| 9.9   | Etapy dopasowania .....                                                             | 251 |
| 9.9.1 | Etap 1. Dopasowanie dosłownie .....                                                 | 252 |
| 9.9.2 | Etap 2. Dopasowanie dosłowne, ale z tzw. trywialną konwersją .....                  | 252 |
| 9.9.3 | Etap 3. Dopasowanie z awansem .....                                                 | 253 |
| 9.9.4 | Etap 4. Próba dopasowania za pomocą konwersji standardowych .....                   | 254 |
| 9.9.5 | Etap 5. Próba dopasowania z użyciem konwersji zdefiniowanych przez użytkownika..... | 256 |
| 9.9.6 | Etap 6. Próba dopasowania do funkcji z wielokropkiem .....                          | 256 |
| 9.10  | Dopasowywanie wywołań z kilkoma argumentami .....                                   | 256 |

## **Tom II**

## **10 Klasy ..... 259**

|      |                                          |     |
|------|------------------------------------------|-----|
| 10.1 | Typy definiowane przez użytkownika ..... | 259 |
| 10.2 | Składniki klasy .....                    | 261 |
| 10.3 | Składnik będący obiektem .....           | 263 |
| 10.4 | Enkapsulacja .....                       | 263 |
| 10.5 | Ukrywanie informacji .....               | 264 |
| 10.6 | Klasa a obiekt .....                     | 267 |



|         |                                                                                   |     |
|---------|-----------------------------------------------------------------------------------|-----|
| 10.7    | Funkcje składowe .....                                                            | 270 |
| 10.7.1  | Posługiwanie się funkcjami składowymi .....                                       | 270 |
| 10.7.2  | Definiowanie funkcji składowych .....                                             | 271 |
| 10.8    | 10.8 Jak to właściwie jest ? ( <i>this</i> ) .....                                | 276 |
| 10.9    | 10.9 Odwołanie się do publicznych danych składowych .....                         | 277 |
| 10.10   | 10.10 Zasłanianie nazw .....                                                      | 278 |
| 10.11   | 10.11 Przeładowanie i zasłonięcie równocześnie .....                              | 281 |
| 10.12   | 10.12 Przesyłanie do funkcji argumentów będącymi obiektami .....                  | 282 |
| 10.12.1 | Przesyłanie obiektu przez wartość .....                                           | 282 |
| 10.12.2 | Przesyłanie przez referencję .....                                                | 285 |
| 10.13   | 10.13 Konstruktor – pierwsza wzmianka .....                                       | 286 |
| 10.14   | 10.14 Destraktor – pierwsza wzmianka .....                                        | 291 |
| 10.15   | 10.15 Składnik statyczny .....                                                    | 295 |
| 10.16   | 10.16 Statyczna funkcja składowa .....                                            | 299 |
| 10.17   | 10.17 Do czego może nam się przydać składnik statyczny w klasie? .....            | 302 |
| 10.18   | 10.18 Funkcje składowe typu <code>const</code> oraz <code>volatile</code> .....   | 303 |
| 10.18.1 | Przeładowanie a funkcje składowe <code>const</code> i <code>volatile</code> ..... | 306 |

## **11 Funkcje zaprzyjaźnione .....307**

## **12 Struktury, Unie, Pola bitowe .....318**

|        |                                 |     |
|--------|---------------------------------|-----|
| 12.1   | 12.1 Struktura .....            | 318 |
| 12.2   | 12.2 Unia.....                  | 319 |
| 12.2.1 | 12.2.1 Inicjalizacja unii ..... | 321 |
| 12.2.2 | 12.2.2 Unia anonimowa .....     | 321 |
| 12.3   | 12.3 Pola bitowe .....          | 323 |

## **13 Zagnieżdżona definicja klasy .....328**

|      |                                    |     |
|------|------------------------------------|-----|
| 13.1 | 13.1 Lokalna definicja klasy ..... | 331 |
| 13.2 | 13.2 Lokalne nazwy typów .....     | 334 |

## **14 Konstruktory i Destruktory .....336**

|        |                                                                             |     |
|--------|-----------------------------------------------------------------------------|-----|
| 14.1   | 14.1 Konstruktor .....                                                      | 336 |
| 14.1.1 | 14.1.1 Przykład programu zawierającego klasę z konstruktorami .....         | 337 |
| 14.2   | 14.2 Kiedy i jak wywoływany jest konstruktor.....                           | 343 |
| 14.2.1 | 14.2.1 Konstruowanie obiektów lokalnych .....                               | 343 |
| 14.2.2 | 14.2.2 Konstruowanie obiektów globalnych .....                              | 343 |
| 14.2.3 | 14.2.3 Konstrukcja obiektów tworzonych operatorem <code>new</code> .....    | 344 |
| 14.2.4 | 14.2.4 Jawne wywołanie konstruktora .....                                   | 345 |
| 14.2.5 | 14.2.5 Dalsze sytuacje, gdy pracuje konstruktor.....                        | 346 |
| 14.3   | 14.3 Destraktor .....                                                       | 347 |
| 14.4   | 14.4 Konstruktor domniemany .....                                           | 349 |
| 14.5   | 14.5 Lista inicjalizacyjna konstruktora .....                               | 350 |
| 14.6   | 14.6 Konstrukcja obiektu, którego składnikiem jest obiekt innej klasy ..... | 353 |
| 14.7   | 14.7 Konstruktory nie-publiczne ? .....                                     | 359 |
| 14.8   | 14.8 Konstruktor kopiujący (albo inicjator kopiujący) .....                 | 361 |
| 14.8.1 | 14.8.1 Przykład klasy z konstruktorem kopiującym .....                      | 363 |
| 14.8.2 | 14.8.2 Konstruktor kopiujący gwarantujący nietykalność .....                | 370 |
| 14.8.3 | 14.8.3 Współodpowiedzialność .....                                          | 371 |
| 14.8.4 | 14.8.4 Konstruktor kopiujący generowany automatycznie .....                 | 372 |

|        |                                                   |     |
|--------|---------------------------------------------------|-----|
| 14.8.5 | Kiedy konstruktor kopiujący jest niezbędny? ..... | 372 |
|--------|---------------------------------------------------|-----|

## **15 Tablice obiektów ..... 377**

|        |                                                         |     |
|--------|---------------------------------------------------------|-----|
| 15.1   | Tablica obiektów definiowana operatorem new .....       | 379 |
| 15.2   | Inicjalizacja tablic obiektów .....                     | 380 |
| 15.2.1 | Inicjalizacja tablic obiektów będących agregatami ..... | 380 |
| 15.2.2 | Inicjalizacja tablic nie będących agregatami .....      | 383 |
| 15.2.3 | Inicjalizacja tablic tworzonych w zapasie pamięci ..... | 386 |

## **16 Wskaźnik do składników klasy ..... 388**

|      |                                                      |     |
|------|------------------------------------------------------|-----|
| 16.1 | Wskaźniki zwykłe - repetytorium .....                | 388 |
| 16.2 | Wskaźnik do pokazywania na składnik-daną .....       | 390 |
| 16.3 | Wskaźnik do funkcji składowej .....                  | 394 |
| 16.4 | Tablica wskaźników do danych składowych klasy .....  | 396 |
| 16.5 | Tablica wskaźników do funkcji składowych klasy ..... | 397 |
| 16.6 | Wskaźniki do składników statycznych .....            | 398 |

## **17 Konwersje ..... 399**

|        |                                                                       |     |
|--------|-----------------------------------------------------------------------|-----|
| 17.1   | Sformułowanie problemu .....                                          | 399 |
| 17.2   | Konstruktor jako konwerter .....                                      | 401 |
| 17.3   | Funkcja konwertująca – operator konwersji .....                       | 404 |
| 17.4   | Który wariant konwersji wybrać ? .....                                | 410 |
| 17.5   | Sytuacje, w których zachodzi konwersja .....                          | 412 |
| 17.6   | Zapis jawnego wywołania konwersji typów .....                         | 413 |
| 17.6.1 | Advocatus zapisu przypominającego: „wywołanie funkcji” .....          | 414 |
| 17.6.2 | Advocatus zapisu: „rzutowanie” .....                                  | 414 |
| 17.7   | Niecałkiem dobrane małżeństwa, czyli konwersje przy dopasowaniu ..... | 415 |
| 17.8   | Kilka rad dotyczących konwersji .....                                 | 420 |

## **18 Przeładowanie operatorów ..... 422**

|         |                                                                             |     |
|---------|-----------------------------------------------------------------------------|-----|
| 18.1    | Przeładowanie operatorów – definicja i trochę teorii .....                  | 424 |
| 18.2    | Moje zabawki .....                                                          | 428 |
| 18.3    | Funkcja operatorowa jako funkcja składowa .....                             | 430 |
| 18.4    | Funkcja operatorowa nie musi być przyjacielem klasy .....                   | 433 |
| 18.5    | Operatory predefiniowane .....                                              | 434 |
| 18.6    | Argumentowość operatorów .....                                              | 434 |
| 18.7    | Operatory jednoargumentowe .....                                            | 435 |
| 18.8    | Operatory dwuargumentowe .....                                              | 438 |
| 18.8.1  | Przykład na przeładowanie operatora dwuargumentowego .....                  | 438 |
| 18.8.2  | Przemienność .....                                                          | 440 |
| 18.9    | Przykład zupełnie niematematyczny .....                                     | 441 |
| 18.10   | Cztery operatory, które muszą być niestatycznymi funkcjami składowymi ..... | 450 |
| 18.11   | Operator przypisania = .....                                                | 451 |
| 18.11.1 | Przykład na przeładowanie operatora przypisania .....                       | 455 |
| 18.11.2 | Jak to opowiedzieć potocznie? .....                                         | 461 |
| 18.11.3 | Kiedy operator przypisania nie jest generowany automatycznie .....          | 463 |
| 18.12   | Operator [ ] .....                                                          | 464 |
| 18.13   | Operator ( ) .....                                                          | 468 |
| 18.14   | Operator -> .....                                                           | 470 |
| 18.15   | Operator new .....                                                          | 477 |



|       |                                                                                           |     |
|-------|-------------------------------------------------------------------------------------------|-----|
| 18.16 | Operator delete .....                                                                     | 479 |
| 18.17 | Operatory postinkrementacji i postdekrementacji, czyli koniec z niesprawiedliwością ..... | 480 |
| 18.18 | Rady praktyczne dotyczące przeładowania .....                                             | 482 |
| 18.19 | Pojedynek: Operator jako funkcja składowa, czy globalna .....                             | 484 |
| 18.20 | Zasłona spada, czyli tajemnica operatora << .....                                         | 486 |
| 18.21 | Rzut oka wstecz .....                                                                     | 492 |

## Tom III

|           |                                                                                         |            |
|-----------|-----------------------------------------------------------------------------------------|------------|
| <b>19</b> | <b>Dziedziczenie .....</b>                                                              | <b>495</b> |
| 19.1      | Istota dziedziczenia .....                                                              | 495        |
| 19.2      | Dostęp do składników .....                                                              | 498        |
| 19.2.1    | Prywatne składniki klasy podstawowej .....                                              | 498        |
| 19.2.2    | Nieprywatne składniki klasy podstawowej .....                                           | 500        |
| 19.2.3    | Klasa pochodna też decyduje .....                                                       | 501        |
| 19.2.4    | Po znajomości, czyli udostępnianie wybiórcze .....                                      | 503        |
| 19.3      | Czego się nie dziedziczy .....                                                          | 504        |
| 19.3.1    | Niedziedziczenie konstruktorów .....                                                    | 504        |
| 19.3.2    | Niedziedziczenie operatora przypisania .....                                            | 505        |
| 19.3.3    | Niedziedziczenie destruktora .....                                                      | 505        |
| 19.4      | Dziedziczenie kilkupokoleniowe .....                                                    | 506        |
| 19.5      | Dziedziczenie – doskonałe narzędzie programowania .....                                 | 507        |
| 19.6      | Kolejność wywoływania konstruktorów .....                                               | 509        |
| 19.7      | Przypisanie i inicjalizacja obiektów w warunkach dziedziczenia .....                    | 515        |
| 19.7.1    | Klasa pochodna nie definiuje swojego operatora przypisania .....                        | 515        |
| 19.7.2    | Klasa pochodna nie definiuje swojego konstruktora kopiującego .....                     | 516        |
| 19.7.3    | Inicjalizacja i przypisywanie według obiektu wzorcowego będącego const .....            | 517        |
| 19.7.4    | Definiowanie konstruktora kopiującego i operatora przypisania dla klasy pochodnej ..... | 517        |
| 19.8      | Dziedziczenie wielokrotne .....                                                         | 522        |
| 19.8.1    | Konstruktor klasy pochodnej przy wielokrotnym dziedziczeniu .....                       | 524        |
| 19.8.2    | Ryzyko wieloznaczności przy dziedziczeniu .....                                         | 526        |
| 19.8.3    | Bliższe pokrewieństwo usuwa wieloznaczność .....                                        | 528        |
| 19.8.4    | Poszlaki .....                                                                          | 528        |
| 19.9      | Pojedynek: Dziedziczenie klasy contra zwieranie obiektów składowych .....               | 529        |
| 19.10     | Konwersje standardowe przy dziedziczeniu .....                                          | 531        |
| 19.10.1   | Panorama korzyści .....                                                                 | 535        |
| 19.10.2   | Czego robić nie można .....                                                             | 537        |
| 19.10.3   | Konwersje standardowe wskaźników do pokazywania we wnętrzu klasy .....                  | 540        |
| 19.11     | Wirtualne klasy podstawowe .....                                                        | 542        |
| 19.11.1   | Publiczne i prywatne dziedziczenie tej samej klasy wirtualnej .....                     | 545        |
| 19.11.2   | Uwagi o konstrukcji i inicjalizacji w wypadku klas wirtualnych .....                    | 546        |
| 19.11.3   | Dominacja klas wirtualnych .....                                                        | 549        |
| <b>20</b> | <b>Funkcje wirtualne .....</b>                                                          | <b>552</b> |
| 20.1      | Polimorfizm .....                                                                       | 559        |

|       |                                                                                     |     |
|-------|-------------------------------------------------------------------------------------|-----|
| 20.2  | Dalsze szczegóły .....                                                              | 562 |
| 20.3  | Wczesne i późne wiązanie .....                                                      | 565 |
| 20.4  | Kiedy dla wywołań funkcji wirtualnych mimo wszystko zachodzi wczesne wiązanie ..... | 566 |
| 20.5  | Kulisy białej magii, czyli: Jak to jest zrobione ? .....                            | 568 |
| 20.6  | Funkcja wirtualna, a mimo to inline .....                                           | 570 |
| 20.7  | Pojedynek – funkcje przeładowane contra funkcje wirtualne .....                     | 570 |
| 20.8  | Klasy abstrakcyjne .....                                                            | 571 |
| 20.9  | Destruktor? to najlepiej wirtualny! .....                                           | 578 |
| 20.10 | Co prawda konstruktor nie może być wirtualny, ale... ..                             | 583 |
| 20.11 | Finis coronat opus.....                                                             | 588 |

## **21 Operacje Wejścia / Wyjścia ..... 590**

|         |                                                                                     |     |
|---------|-------------------------------------------------------------------------------------|-----|
| 21.1    | Biblioteka iostream .....                                                           | 591 |
| 21.2    | Strumień .....                                                                      | 592 |
| 21.3    | Strumienie predefiniowane .....                                                     | 593 |
| 21.4    | Operatory >> i << .....                                                             | 594 |
| 21.5    | Domniemania w pracy strumieni predefiniowanych .....                                | 595 |
| 21.6    | Uwaga na priorytet .....                                                            | 598 |
| 21.7    | Operatory << oraz >> definiowane przez użytkownika .....                            | 600 |
| 21.7.1  | Operatorów wstawiania i wyjmowania ze strumienia - nie dziedziczy się .....         | 604 |
| 21.7.2  | Operatory wstawiania i wyjmowania nie mogą być wirtualne. Niestety. ....            | 606 |
| 21.8    | Sterowanie formatem .....                                                           | 607 |
| 21.9    | Flagi stanu formatowania .....                                                      | 608 |
| 21.9.1  | Znaczenie poszczególnych flag sterowania formatem .....                             | 609 |
| 21.10   | Sposoby zmiany trybu (reguł) formatowania .....                                     | 612 |
| 21.10.1 | Zmiana sposobu formatowania funkcjami <code>setf</code> , <code>unsetf</code> ..... | 614 |
| 21.10.2 | Wygodniejsze funkcje do zmiany stanu formatowania. ....                             | 616 |
| 21.11   | Manipulatory .....                                                                  | 621 |
| 21.11.1 | Manipulatory bezargumentowe .....                                                   | 622 |
| 21.11.2 | Manipulatory parametryzowane .....                                                  | 624 |
| 21.11.3 | Definiowanie swoich manipulatorów .....                                             | 628 |
| 21.12   | Nieformatowane operacje wejścia/wyjścia .....                                       | 631 |
| 21.13   | Omówienie funkcji wyjmujących ze strumienia.....                                    | 633 |
| 21.13.1 | Funkcje do pracy ze znakami i stringami .....                                       | 633 |
| 21.13.2 | Wczytywanie binarne – funkcja <code>read</code> .....                               | 638 |
| 21.13.3 | Funkcja <code>ignore</code> .....                                                   | 639 |
| 21.13.4 | Pożyteczne funkcje pomocnicze .....                                                 | 640 |
| 21.13.5 | Funkcje wstawiające do strumienia .....                                             | 643 |
| 21.14   | Operacje we/wy na plikach .....                                                     | 644 |
| 21.14.1 | Otwieranie i zamykanie strumienia .....                                             | 646 |
| 21.15   | Błędy w trakcie pracy strumienia .....                                              | 650 |
| 21.15.1 | Flagi stanu błędu strumienia .....                                                  | 650 |
| 21.15.2 | Funkcje do pracy na flagach błędu .....                                             | 651 |
| 21.15.3 | Kilka udogodnień .....                                                              | 652 |
| 21.15.4 | Bardziej wyszukane operacje na flagach błędu strumienia .....                       | 654 |
| 21.15.5 | Trzy plagi - czyli „gotowiec” jak radzić sobie z błędami .....                      | 657 |



|         |                                                                          |     |
|---------|--------------------------------------------------------------------------|-----|
| 21.16   | Przykład programu pracującego na plikach .....                           | 660 |
| 21.17   | Wybór miejsca czytania lub pisania w pliku .....                         | 662 |
| 21.17.1 | Funkcje składowe informujące o pozycji wskaźników .....                  | 663 |
| 21.17.2 | Wybrane funkcje składowe do pozycjonowania wskaźników .....              | 664 |
| 21.18   | Przykład większego programu .....                                        | 665 |
| 21.19   | Tie – harmonijna praca dwóch strumieni .....                             | 672 |
| 21.20   | Attach – zmiana koryta strumienia .....                                  | 674 |
| 21.21   | Dlaczego tak nie lubimy biblioteki stdio? .....                          | 675 |
| 21.22   | Niektóre aspekty współżycia biblioteki stdio z biblioteką iostream ..... | 677 |
| 21.23   | Formatowanie wewnętrzne – Operacje wyjścia .....                         | 679 |
| 21.23.1 | Mechanizm automatycznej rezerwacji miejsca .....                         | 682 |
| 21.23.2 | Anonimowy strumień wyjściowy .....                                       | 685 |
| 21.24   | Formatowanie wewnętrzne – operacje wejścia .....                         | 685 |
| 21.24.1 | Przykładowe zastosowanie oraz anonimowy strumień wejściowy .....         | 688 |
| 21.25   | Ożenek: klasy wejściowo – wyjściowe dla formatowania wewnętrznego .....  | 690 |
| 21.26   | Jeszcze o strumieniach anonimowych .....                                 | 691 |

## **22 Projektowanie programów obiektowo orientowanych.....693**

|        |                                                                                         |     |
|--------|-----------------------------------------------------------------------------------------|-----|
| 22.1   | Przegląd kilku technik programowania .....                                              | 694 |
| 22.1.1 | Programowanie liniowe .....                                                             | 694 |
| 22.1.2 | Programowanie proceduralne .....                                                        | 694 |
| 22.1.3 | Programowanie z ukrywaniem danych .....                                                 | 695 |
| 22.1.4 | Programowanie obiektowe - programowanie „bazujące” na obiektach .....                   | 695 |
| 22.1.5 | Programowanie Obiektowo Orientowane (OO) .....                                          | 696 |
| 22.2   | O wyższości programowania obiektowo orientowanego nad Świętami Wielkiej Nocy .....      | 696 |
| 22.3   | Obiektowo Orientowane: Projektowanie .....                                              | 699 |
| 22.4   | Praktyczne wskazówki dotyczące projektowania programu techniką OO .....                 | 701 |
| 22.4.1 | Rekonesans – czyli rozpoznanie zagadnienia .....                                        | 701 |
| 22.4.2 | Faza projektowania .....                                                                | 702 |
| 22.4.3 | Etap 1: Identyfikacja zachowań systemu .....                                            | 703 |
| 22.4.4 | Etap 2: Identyfikacja obiektów (klas obiektów) .....                                    | 704 |
| 22.4.5 | Etap 3: Usystematyzowanie klas obiektów .....                                           | 705 |
| 22.4.6 | Etap 4: Określenie wzajemnych zależności klas .....                                     | 707 |
| 22.4.7 | Etap 5: Składanie modelu. Określanie sekwencji działań obiektów i cykli życiowych ..... | 709 |
| 22.5   | Faza implementacji .....                                                                | 710 |
| 22.6   | Przykład projektowania .....                                                            | 710 |
| 22.7   | Faza: Rozpoznanie naszego zagadnienia .....                                             | 711 |
| 22.8   | Faza: Projektowanie .....                                                               | 715 |
| 22.8.1 | Etap 1 – Identyfikacja zachowań naszego systemu .....                                   | 715 |
| 22.8.2 | Etap 2 – Identyfikacja klas obiektów, z którymi mamy do czynienia .....                 | 716 |
| 22.8.3 | Etap 3 - Usystematyzowanie klas obiektów z występujących w naszym systemie .....        | 719 |
| 22.8.4 | Etap 4 - Określenie wzajemnych zależności klas .....                                    | 721 |
| 22.8.5 | Etap 5 - Składamy model naszego systemu .....                                           | 723 |
| 22.9   | Implementacja modelu naszego systemu .....                                              | 727 |
| 22.10  | Symfonia C++, Coda .....                                                                | 734 |
| 22.11  | Posłowie .....                                                                          | 734 |