

Komputery, ludzie i programowanie

„Specjalizowanie się pozostawmy owadom.”

— R. A. Heinlein

W tym rozdziale opisujemy powody, dla których naszym zdaniem programowanie jest ważne, interesujące i zabawne. Prezentujemy też kilka podstawowych myśli i ideałów. Mamy nadzieję obalić kilka popularnych mitów na temat programowania i programistów. Rozdział ten najlepiej przejrzeć teraz i wrócić do niego później, gdy walcząc z jakimś problemem, będziesz się zastanawiać nad sensem swoich działań.

1.1. Wstęp

1.2. Oprogramowanie

1.3. Ludzie

1.4. Informatyka

1.5. Komputery są wszędzie

1.5.1. Komputery z ekranem i bez

1.5.2. Transport

1.5.3. Telekomunikacja

1.5.4. Medycyna

1.5.5. Informacja

1.5.6. Sięgamy w kosmos

1.5.7. I co z tego

1.6. Ideały dla programistów

1.1. Wstęp

Podobnie jak w większości dziedzin, z nauką programowania wiąże się problem jajka i kury. Chcemy zacząć, ale pragniemy też wiedzieć, czemu to, czego chcemy się nauczyć, jest ważne. Zależy nam, aby zdobyć praktyczne umiejętności, ale chcemy też mieć pewność, że nie jest to tylko jakaś przejściowa moda. Nie chcemy zmarnować czasu, ale nie chcemy też być znużeni nadmiernym entuzjazmem i moralizowaniem. Przeczytaj tyle z tego rozdziału, ile wyda Ci się interesujące. Możesz do niego wrócić później, aby odświeżyć sobie pamięć w kwestii tego, dlaczego techniczne szczegóły mają znaczenie także poza murami szkoły.

Rozdział ten zawiera nasze osobiste wyznania na temat tego, co według nas jest interesujące i ważne w programowaniu. Wyjaśniamy tu, co trzyma nas w tej dziedzinie po tylu latach pracy. Czytając ten rozdział, poznasz potencjalne cele, do których można dążyć, oraz dowiesz się, jakiego rodzaju osobą jest programista. Książki dla początkujących muszą zawierać dużo podstawowych informacji. W rozdziale tym porzucamy techniczne szczegóły i próbujemy przedstawić ogólny obraz wyjaśniający, czemu warto zainteresować się programowaniem. Jaką rolę odgrywa w naszej cywilizacji? Gdzie programista może się przysłużyć, aby móc być z siebie dumny? Jakie jest miejsce programowania w szerszej dziedzinie wytwarzania, wdrażania i konserwowania oprogramowania? Gdzie umieścić programowanie, gdy ludzie rozmawiają o informatyce, inżynierii oprogramowania lub technologiach informacyjnych? Co robi programista? Jakie umiejętności cechują dobrego programistę?

Dla studenta najważniejszym czynnikiem motywującym do zrozumienia jakiegoś pojęcia czy jakiejś techniki albo treści rozdziału może być chęć zdania egzaminu na dobrą ocenę — ale w nauce chodzi o coś więcej! Dla osoby pracującej w firmie wytwarzającej oprogramowanie czynnikiem motywującym do zrozumienia jakiegoś pojęcia czy jakiejś techniki albo treści rozdziału może być chęć znalezienia czegoś, co pomoże w pracy nad aktualnym projektem i nie zdenerwuje szefa, który kontroluje listę płac, decyduje o awansach i zwolnieniach — ale w nauce chodzi o coś więcej! Pracujemy wydajniej, jeśli czujemy, że to, co robimy, w jakiś sposób, choćby w niewielkim stopniu, przyczynia się do polepszenia świata. Dla zadań, które wykonujemy przez wiele lat (tych rzeczy, z których składają się zawody i profesjonalne kariery), najważniejsze są ideały i abstrakcyjne pomysły.



Nasza cywilizacja opiera się na oprogramowaniu. Dlatego polepszanie jego jakości i znajdowanie nowych zastosowań dla programów to dwa sposoby na to, aby pomóc innym w polepszeniu komfortu ich życia. Programowanie odgrywa w tym najważniejszą rolę.

1.2. Oprogramowanie

Dobre oprogramowanie jest niewidoczne. Nie można go zobaczyć, poczuć, zważyć ani dotknąć ręką. **Oprogramowanie** to zestaw programów, które działają na komputerze. Czasami widać ten komputer. Często jednak widać tylko to, co go zawiera, np. telefon, aparat fotograficzny, piekarnik do chleba, samochód albo turbina wiatrowa. Widzimy efekt działania oprogramowania. Jeśli nie wykonuje ono dobrze swoich obowiązków, może nas to denerwować, a nawet zrobić nam krzywdę. Możemy się irtować lub zostać zranieni, jeśli to, co ono robi, nie odpowiada naszym potrzebom.

Ile jest komputerów na świecie? Nie wiadomo. Na pewno są to liczby rzędu miliardów. Może być ich więcej niż ludzi. Zaliczają się do nich serwery, komputery biurkowe, laptopy, tablety, smartfony i komputery wbudowane w gadzety.

Z ilu komputerów korzystamy każdego dnia? W moim samochodzie jest ich ponad 30, dwa w telefonie komórkowym, jeden w odtwarzaczu MP3 i jeden w aparacie fotograficznym. Mam jeszcze laptopa (przy jego użyciu napisałem tekst, który czytasz) oraz komputer stacjonarny. Urządzenie klimatyzacyjne, które kontroluje temperaturę i wilgotność powietrza, także zawiera prosty komputer. Jeden steruje windą naszego wydziału. Jeśli masz nowy telewizor, to w nim też jest przynajmniej jeden komputer. Surfuując w internecie, bezpośrednio komunikujesz się z dziesiątkami — a może nawet setkami — serwerów z całego świata. Wszystko to odbywa się za pośrednictwem systemu telekomunikacyjnego składającego się z tysięcy komputerów — centrali telefonicznych, routerów itp.

Nie, nie mam na tylnym siedzeniu samochodu 30 laptopów! Chodzi mi o to, że większość komputerów nie wygląda tak, jak sobie to najczęściej wyobrażamy (monitor, klawiatura, mysz itd.). Są niewielkimi wstawkami wbudowanymi w rozmaite urządzenia, których używamy. Mój samochód nie ma zatem nic, co wygląda jak komputer, nawet modnych ostatnio gadżetów typu ekran wyświetlający mapę czy wskazujący kierunek jazdy. Ale w silniku jest sporo komputerów, które sterują takimi mechanizmami jak wtrysk paliwa i układ kontroli temperatury. Wspomaganie kierowcy korzysta z przynajmniej jednego komputera, kilka można znaleźć w radiu i układzie bezpieczeństwa. Podejrzewam nawet, że przyciski otwierania i zamykania okien są sterowane komputerem. W nowszych modelach są nawet komputery, które cały czas kontrolują ciśnienie powietrza w oponach.

Ile komputerów potrzebujesz w swoich codziennych czynnościach? Jesz. Jeśli mieszkasz w nowoczesnym mieście, zdobycie pożywienia jest poważnym przedsięwzięciem wymagającym cudów planowania, transportu i przechowywania zapasów. Zarządzanie sieciami dystrybucji jest oczywiście wspomagane komputerowo, podobnie jak systemy komunikacyjne, które je łączą. Nowoczesne rolnictwo jest wysoce skomputeryzowane. Obok obory znajdują się komputery kontrolujące stan stada (wiek osobników, zdrowie, wydajność mleczna itp.). Coraz więcej komputerów pojawia się w maszynach rolniczych, a liczba formularzy wymaganych przez różne agencje rządowe może doprowadzić każdego uczciwego rolnika do płaczu. Jeśli stanie się coś złego, możesz o tym przeczytać w gazecie. Oczywiście artykuły do gazet pisane są i przygotowywane do czytania przy użyciu komputerów oraz (jeśli czytasz je w wersji „martwego drzewa”) drukuje się je za pomocą urządzeń, którymi sterują komputery — często po przesłaniu drogą elektroniczną do drukarni. Książki wytwarza się w ten sam sposób. Jeśli chcesz gdzieś dojechać, wiedz, że ruch uliczny jest kontrolowany przez komputery, które (zazwyczaj na próżno) próbują zapobiegać powstawaniu korków ulicznych. Wolisz pociąg? Też są skomputeryzowane. Niektóre nawet jeżdżą bez maszynisty, a znajdujące się w nich podsystemy, takie jak ogłaszanie, hamowanie i wydawanie biletów, zawierają mnóstwo komputerów. Do największych użytkowników komputerów należy dzisiejsza branża rozrywkowa (muzyka, film, telewizja, występy sceniczne). Nawet w filmach innych niż kreskówki wykorzystuje się bardzo dużo animacji. Muzyka i fotografia także są cyfrowe (tzn. są tworzone z wykorzystaniem komputerów) i komputery wykorzystuje się zarówno do ich zapisu, jak i przesyłania. Jeśli zachorujesz, lekarz zleci badania medyczne, których wykonanie wiąże się z użyciem wielu komputerów. Także większość aparatury medycznej, z którą będziesz mieć kontakt, jeśli wylądujesz w szpitalu, wykorzystuje komputery. Jeśli nie mieszkasz w chacie w środku lasu bez dostępu do urządzeń elektrycznych (włącznie z żarówkami), wykorzystujesz energię. Ropę wykrywa się, wydobywa, przetwarza i rozprowadza przy użyciu systemów, które na każdym etapie wykorzystują komputery — od maszyn wiertniczych, których wiertła tkwią głęboko w ziemi, po dystrybutor na stacji benzynowej, na której tankujesz. Jeśli za paliwo zapłacisz kartą kredytową,

znowu korzystasz z usług całej rzeszy komputerów. To samo można powiedzieć o energii pozyskiwanej z węgla, gazu, słońca i wiatru.

Wszystkie przedstawione do tej pory przykłady mają charakter czynnościowy — bezpośrednio dotyczą tego, co człowiek robi. Z tym luźno wiąże się ważna i zarazem interesująca dziedzina projektowania. Ubrania, które masz na sobie, telefon, przez który rozmawiasz, i automat, który parzy Twoją ulubioną kawę, zostały zaprojektowane i wyprodukowane przy użyciu komputerów. Doskonała jakość nowoczesnych soczewek fotograficznych oraz niezwykle kształty rzeczy codziennego użytku prawie w stu procentach swoją jakość zawdzięczają komputerom. Rzemieślnicy, projektanci, artyści i inżynierzy, którzy kształtują nasze środowisko, zostali uwolnieni od wielu fizycznych ograniczeń, które kiedyś uważano za niemożliwe do przeskoczenia. Jeśli zachorujesz, będziesz brać leki, które zostały zaprojektowane za pomocą komputerów.

W końcu sama nauka w dużym stopniu wykorzystuje komputery. Teleskopy zgłębiające tajemnice najodleglejszych gwiazd nie mogłyby powstać bez komputerów, nie wspominając nawet o sterowaniu nimi i analizowaniu ogromnych ilości danych, które produkują. Pojedynczy biolog może nie być za bardzo skomputeryzowany (pomijając oczywiście takie rzeczy jak aparat fotograficzny, dyktafon, telefon komórkowy itp.), ale kiedy pracuje w laboratorium, musi jakoś zapisywać i analizować dane, porównywać je z modelami komputerowymi oraz kontaktować się z innymi badaczami. W nowoczesnej chemii i biologii — włącznie z badaniami z zakresu medycyny — komputery odgrywają tak dużą rolę, że większość ludzi nawet dzisiaj sobie tego nie uświadamia. Ludzki genom został zsekwencjonowany przez komputery. Mówiąc dokładniej, zsekwencjonowali go ludzie posługujący się komputerami. Wszystkie te przykłady pokazują, że dzięki komputerom wiele rzeczy, które kiedyś były niemożliwe, teraz można wykonać.

Każdy z tych komputerów potrzebuje oprogramowania. Bez niego byłyby to tylko drogie kawałki krzemu, metalu i plastiku: odbojniki do drzwi, kotwice statków i grzejniki. Każdy wiersz kodu został napisany przez człowieka. Każda linijka kodu powstała wskutek myśli ludzkiej, która przecież nie zawsze jest idealna. Zdziwiłoby, że to wszystko działa! Mówimy o miliardach wierszy kodu (tekstu programu) napisanego w setkach języków programowania. Zmuszenie tego wszystkiego do działania wymagało niesłychanego wysiłku oraz niewyobrażalnej wręcz liczby rozmaitych umiejętności. Chcemy, aby każdy gadżet, którego używamy, był jak najlepszy. Wyobraź sobie dowolne urządzenie, którego często używasz. Co chciałbyś w nim poprawić? Między innymi chcemy, aby nasze zabawki były mniejsze (lub większe), szybsze, bardziej niezawodne, miały więcej funkcji i większą pojemność, lepiej wyglądały, były łatwiejsze w użyciu i tańsze. Jest spore prawdopodobieństwo, że poprawki, o których pomyślałeś, wymagałyby zmiany oprogramowania.

1.3. Ludzie



Komputery są budowane przez ludzi dla ludzi. Są to bardzo wszechstronne urządzenia. Można je wykorzystywać do wykonywania niewyobrażalnej liczby zadań. Innymi słowy, komputer to tylko kawałek żelastwa, dopóki ktoś — jakiś programista — nie napisze dla niego oprogramowania. Często o tym zapominamy. Jeszcze częściej zapomina się o programistach.

Hollywood i inne podobne źródła dezinformacji wywodzące się z kultury masowej przykleiły programistom negatywną etykietkę. Każdy na przykład zna obraz samotnego, grubego i brzydkiego maniaka komputerowego, który nie potrafi żyć między ludźmi i ma obsesję na punkcie gier oraz włamywania się do komputerów innych ludzi. Taki osobnik (najczęściej mężczyzna) może tak samo chcieć zniszczenia świata, jak i uratowania go przed złem. Oczywiście

w realnym świecie istnieją mniej wyraziste wersje takich karykatur, ale naszym zdaniem nie występują w świecie programistów częściej niż wśród prawników, policjantów, sprzedawców samochodów, dziennikarzy, artystów czy polityków.

Pomyśl o znanych Ci programach, których używasz na co dzień. Myślisz, że utworzył je tajemniczy osobnik siedzący w ciemnym pomieszczeniu? Oczywiście nie! Produkcja wysokiej jakości oprogramowania, skomputeryzowanych gadżetów i systemów wymaga ścisłej współpracy dziesiątek, setek a nawet tysięcy ludzi, z których każdy ma swoją wyraźnie określoną rolę. Są to programiści, projektanci (także oprogramowania), testerzy, kierownicy grup dyskusyjnych, psychologowie eksperymentalni, projektanci interfejsów użytkownika, analitycy, administratorzy systemu, osoby odpowiedzialne za kontakty z klientami, inżynierowie dźwięku, kierownicy projektów, osoby od kontroli jakości, statystycy, inżynierowie interfejsów sprzętowych, inżynierowie wymagań, specjaliści od zabezpieczeń, matematycy, sprzedawcy, osoby zajmujące się rozwiązywaniem problemów, projektanci sieciowi, metodocy, osoby zarządzające narzędziami programistycznymi, osoby opracowujące biblioteki oprogramowania itd. Zakres ról jest ogromny, a do tego dochodzą jeszcze różnice w nazwach stosowanych w różnych firmach. To, co w jednej firmie nazywa się inżynierem, w innych może być programistą, deweloperem, członkiem personelu technicznego lub architektem. W niektórych firmach pozwala się nawet pracownikom samodzielnie dobierać nazwy stanowisk. Nie wszystkie one bezpośrednio dotyczą programowania. Jednak na własne oczy widzieliśmy osoby, których stanowiska nazywały się tak, jak jedno z wymienionych, i których główną funkcją było czytanie lub pisanie kodu. Ponadto programista (pełniący którąkolwiek z tych ról i więcej) może przez krótki okres mieć styczność z ludźmi wielu innych specjalności, np. biologami, projektantami silników, prawnikami, sprzedawcami samochodów, naukowcami z dziedziny medycyny, historykami, geologami, astronautami, inżynierami lotnictwa, kierownikami składów drzewnych, naukowcami pracującymi nad budową raket kosmicznych, pracownikami budującymi tory, po których toczą się kule w grze w kręgle, dziennikarzami i animatorami (lista została opracowana na podstawie doświadczeń własnych). Są też takie osoby, które czasami pracują jako programiści, ale pełnią też inne funkcje w swojej karierze.

Mit samotnie pracującego programisty jest tylko mitem. Ludzie, którzy lubią pracować na własną rękę, często wybierają takie dziedziny, w których jest to łatwiejsze do osiągnięcia. Często gorzko narzekają na to, że im się przeszkadza i że muszą brać udział w spotkaniach. Ci, którzy lubią współpracować z innymi ludźmi, mają łatwiej w programowaniu, ponieważ nowoczesne programowanie jest pracą zespołową. Wynika z tego taki wniosek, że zdolności interpersonalne w programowaniu są znacznie ważniejsze, niż wskazują stereotypowe poglądy. Na krótkiej liście najbardziej pożądanых cech programisty (przy realistycznej definicji **programisty**) znajdują się wysokie zdolności komunikacyjne — z ludźmi z różnych środowisk — w kontaktach nieformalnych, na spotkaniach, drogą pisemną oraz na formalnych prezentacjach. Jesteśmy pewni, że dopóki nie popracujesz nad jednym lub dwoma projektami, nie będziesz miał pojęcia, co to tak naprawdę jest programowanie oraz czy to Ci się podoba czy nie. Wśród wielu rzeczy, które podobają nam się w naszej pracy programistów, ważne miejsce zajmują spotkania z miłymi i ciekawymi ludźmi oraz odwiedzanie rozmaitych miejsc, do których czasami musimy się udać.

Jednym z wniosków, które można z tego wyciągnąć, jest to, że do wyprodukowania wysokiej jakości programu potrzebni są ludzie o różnych zdolnościach, zainteresowaniach i nawykach. Od nich zależy jakość naszego życia — czasami nawet samo nasze życie. Nie ma człowieka, który mógłby spełniać wszystkie wymienione wymagania. Nikt rozsądny nie chciałby każdej z tych ról. Chcę powiedzieć, że masz o wiele więcej opcji do wyboru, niż Ci się wydaje.

Nie musisz oczywiście dokonywać jednego konkretnego wyboru. Jako odrębna jednostka będziesz dryfować w kierunku dziedzin, w których najlepiej sprawdzają się Twoje umiejętności, talenty i zainteresowania.

Mówimy o programistach i programowaniu, ale oczywiście programowanie to tylko część całego obrazu. Ludzie projektujący okręty lub telefony nie uważają się za programistów. Programowanie to ważny element procesu wytwarzania oprogramowania, ale nie jedyny. Analogicznie do większości produktów, programowanie jest ważnym składnikiem procesu produkcyjnego, ale zdecydowanie nie jedynym.



Nie wychodzimy z założenia, że chcesz zostać profesjonalnym programistą i przez resztę życia pisać kod źródłowy programów. Nawet najlepsi programiści — zwłaszcza ci **najlepsi** — większość czasu spędzają na czymś **innym** niż pisanie kodu. W zrozumienie problemu często trzeba zainwestować dużo czasu i włożyć dużo wysiłku intelektualnego. Ten wysiłek dla wielu programistów jest właśnie tym, co ich pociąga w programowaniu. Wiele programistów ma ukończone takie kierunki studiów, których tradycyjnie nie wiąże się z informatyką. Jeśli na przykład ktoś bierze udział w projekcie oprogramowania do badania genomu, będzie znacznie efektywniejszy, jeśli będzie znał się na biologii molekularnej. Do pracy nad programem do analizy literatury średniowiecznej przydałaby się znajomość tego rodzaju tekstów, a najlepiej także odpowiednich języków. W szczególności osoby z podejściem typu „jedynie, co mnie interesuje, to komputery i programowanie” są niezdolne do porozumiewania się ze współpracownikami reprezentującymi inne dziedziny. Osoba taka nie tylko będzie traciła to, co najlepsze w interakcjach z ludźmi (tj. w życiu), lecz będzie również słabym programistą.

Jakie więc przyjmujemy założenia? Programowanie wymaga wielu ambitnych umiejętności intelektualnych, które są częścią wielu ważnych i interesujących dziedzin techniki. Ponadto programowanie jest podstawową częścią naszego świata, dlatego niezajomość podstaw programowania jest jak niezajomość podstaw fizyki, historii, biologii czy literatury. Osoba, która nie ma pojęcia o programowaniu, musi wierzyć w magię i może być niebezpieczna w przypadku pełnienia wielu technicznych funkcji. Jeśli czytałeś Dilberta, pomyśl o tym szefie ze spiczastą fryzurą jak o takim, którego nie chcesz mieć lub (co gorsza) którym nie chcesz się stać. Poza tym programowanie może być zabawne.

Ale do czego naszym zdaniem mogą przydać Ci się umiejętności programistyczne? Może wykorzystasz je jako jedno z kluczowych narzędzi w dalszych studiach i będziesz pracować, nigdy nie stając się profesjonalnym programistą. Może pracując jako projektant, pisarz, kierownik lub naukowiec, będziesz mieć kontakt z innymi ludźmi na gruncie zawodowym lub osobistym i umiejętności programistyczne będą Ci w tych kontaktach pomocne. Może będziesz profesjonalnie programować w ramach studiów lub obowiązków zawodowych. Nawet jeśli zostaniesz zawodowym programistą, jest bardzo mało prawdopodobne, że nie będziesz robić nic innego poza programowaniem.

Możesz zostać inżynierem interesującym się komputerami lub informatykiem, ale nawet wówczas nie będziesz programować cały czas. Programowanie to sposób wyrażania myśli za pomocą kodu — jedno z narzędzi wspomagających rozwiązywanie problemów. Jest niczym — kompletnym marnowaniem czasu — jeśli nie ma wartych rozpatrzenia pomysłów i problemów wartych rozwiązania.

Tematem tej książki jest programowanie i obiecaliśmy, że nauczymy Cię programować. Dlaczego więc podkreślamy rzeczy niezwiązane z programowaniem oraz kładziemy nacisk na ograniczoną rolę programowania? Dobry programista rozumie rolę kodu i technik programistycznych w projekcie. Dobry programista jest (w większości przypadków) dobrym zawodnikiem

w drużynie, który cały czas myśli nad tym, jaki kod najbardziej przysłuży się projektowi. Wyobraźmy sobie na przykład, że pracowałem nad nowym odtwarzaczem MP3 (który miał zostać dodany do smartfona lub tabletu). Jedyne, co mnie interesowało, to piękno mojego kodu i liczba fajnych funkcji, które udało mi się opracować. Prawdopodobnie uparłbym się, aby mój kod działał na największym i najpotężniejszym dostępnym komputerze. Mógłbym odrzucić teorię kodowania dźwięku, ponieważ to „nie jest programowanie”. Przesiedziałbym cały czas w laboratorium, zamiast wyjść do potencjalnych użytkowników, którzy pewnie i tak mają zły gust, jeśli chodzi o muzykę, oraz pewnie by nie potrafili docenić najnowszych osiągnięć w dziedzinie programowania graficznych interfejsów użytkownika. Jest bardzo prawdopodobne, że projekt zakończyłby się całkowitym fiaskiem. Większy komputer spowodowałby wzrost kosztu odtwarzacza i najprawdopodobniej skróciłby czas działania baterii. Kodowanie jest podstawą zapisu muzyki cyfrowej i pominięcie postępu w tej dziedzinie przyczyniłoby się do zwiększenia wymagań pamięciowych przez każdy utwór (różnice między algorytmami kodowania muzyki są ogromne). Niebranie pod uwagę preferencji użytkowników — bez względu na to, jak dziwne i archaiczne nam się wydają — najczęściej prowadzi do tego, że wybiorą oni inny produkt. Ważną częścią procesu pisania dobrego programu jest zrozumienie potrzeb jego przyszłych użytkowników oraz ograniczeń, jakie te potrzeby nań nakładają. Aby uzupełnić naszą karykaturę programisty, dodamy jeszcze tendencję do spóźniania się z powodu obsesyjnej dbałości o szczegóły i nadmiernej wiary w poprawność słabo przetestowanego kodu. Chcemy, abyś stał się dobrym programistą o szerokim spojrzeniu na to, co wyróżnia dobre oprogramowanie. To jest warunek do usatysfakcjonowania ludzi i klucz do własnej satysfakcji.

1.4. Informatyka

Nawet w najszerzej definicji programowanie postrzega się jako część czegoś większego. Można je traktować jako poddziedzinę informatyki, inżynierii komputerowej, inżynierii oprogramowania, technologii informacyjnej lub jakiegokolwiek innej dziedziny, która ma związek z komputerami. Postrzegamy je jako technologię pobudzającą postęp w takich dziedzinach nauki i inżynierii związanych z komputerami i informacją jak fizyka, biologia, medycyna, historia, literatura i wszystkie inne dziedziny badań akademickich.

Weźmy na przykład informatykę. W „błękitnej księdze” wydanej przez rząd USA w 1995 roku czytamy następującą definicję: „Planowe badanie systemów komputerowych i przetwarzania komputerowego. Wiedza zdobyta w ramach tej dyscypliny pozwala opracowywać teorie działania systemów i metod komputerowych, metodologii projektowych, algorytmów i narzędzi, metod testowania pojęć, metod analizy i weryfikacji oraz reprezentacji wiedzy i implementacji”. Jak można się spodziewać, w Wikipedii znajduje się mniej formalna definicja: „Informatyka to nauka o teoretycznych podstawach informacji i przetwarzania danych oraz ich implementacji i zastosowaniu w systemach komputerowych. Informatyka wykształciła wiele poddziedzin. Niektóre kładą nacisk na uzyskanie specyficznych wyników (np. grafika komputerowa), inne zaś (np. teoria złożoności obliczeniowej) skupiają się na właściwościach problemów obliczeniowych. Jeszcze inne w kręgu zainteresowań stawiają wyzwania związane z implementacją obliczeń. Na przykład teoria języków programowania zajmuje się badaniem różnych sposobów opisu metod obliczeniowych, podczas gdy programowanie komputerowe to rozwiązywanie problemów obliczeniowych za pomocą języków programowania”.



Programowanie to podstawowe narzędzie służące do opracowywania rozwiązań podstawowych i praktycznych problemów, które można przetestować, ulepszyć poprzez wykonanie doświadczeń oraz wykorzystać w praktyce. W programowaniu pomysły i teoria zderzają się z rzeczywistością. Biorąc to pod uwagę, można powiedzieć, że informatyka może stać się dziedziną opartą na doświadczeniach, a nie czystą teorią, i mieć wpływ na świat. W tym kontekście, podobnie jak w wielu innych, należy podkreślić, że programowanie to sposób wyrażania dobrze sprawdzonych praktyk i teorii. Nie można go sprowadzić do kilku sztuczek: wykorzystując jakąkolwiek przestarzałą technikę, po prostu napisz trochę kodu, który natychmiast spełni wszystkie wymagania.

1.5. Komputery są wszędzie

Nikt nie wie wszystkiego, co można wiedzieć o komputerach. W tym podrozdziale przedstawimy tylko kilka wybranych faktów. Może znajdziesz tu coś dla siebie. Przynajmniej może przekonasz się, że zakres zastosowań komputerów — a pośrednio programowania — jest znacznie większy, niż ktokolwiek może sobie wyobrazić.

Dla większości ludzi komputer to nieduża szara skrzynka z podłączonymi monitorem i klawiaturą. Tego typu komputery są najczęściej schowane pod biurkiem i dobrze sprawdzają się w grach, komunikowaniu oraz odtwarzaniu muzyki. Inny rodzaj to tzw. laptopy, których najczęściej używają znudzeni biznesmeni do przeglądania arkuszy kalkulacyjnych, grania w gry komputerowe oraz oglądania filmów. Przedstawiony obraz prezentuje tylko czubek góry lodowej. Większość komputerów działa poza zasięgiem naszych oczu. Wchodzą one w skład systemów, które napędzają naszą cywilizację. Niektóre są rozmiarów pokoju, inne natomiast są mniejsze od monety. Komunikacja z większością najbardziej interesujących komputerów nie odbywa się za pośrednictwem myszy, klawiatury ani żadnego innego podobnego gadżetu.

1.5.1. Komputery z ekranem i bez

Obraz komputera w postaci sporej szarej skrzynki z dołączonymi monitorem i klawiaturą jest bardzo rozpowszechniony i trudno się go pozbyć. Spójrzmy jednak na dwa poniższe komputery:

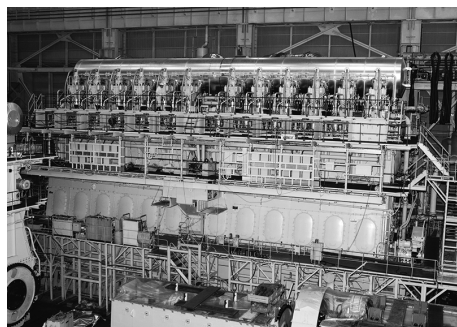


Oba te gadżety (które są przy okazji zegarkami) to przede wszystkim komputery. W istocie przypuszczam, że oba reprezentują ten sam model komputera z różnymi systemami wejścia i wyjścia. Ten po lewej obsługuje niewielki ekran (podobnie jak konwencjonalne komputery ekran monitora), a ten po prawej małe silniki elektryczne, które poruszają wskazówkami

i dyskiem z liczbami oznaczającymi dzień miesiąca. Ich systemy wejścia składają się z czterech przycisków (lepiej widoczne na zegarku po prawej) oraz radiowego urządzenia odbiorczego służącego do synchronizacji z bardzo precyzyjnymi zegarami atomowymi. Większość programów wykorzystywanych do sterowania tymi komputerami jest taka sama.

1.5.2. Transport

Na poniższych ilustracjach widać duży silnik diesla oraz rodzaj okrętu, który można w niego wyposażać:



Pomyślmy, gdzie w tych dwóch przypadkach komputery odegrały kluczową rolę:

- *Projekt* — oczywiście zarówno statek, jak i silnik zostały zaprojektowane przy użyciu komputerów. Lista zastosowań jest praktycznie nieskończona. Zawiera opracowanie rysunków technicznych i inżynierskich, ogólne obliczenia, wizualizację przestrzeni i niektórych części oraz symulację działania niektórych podzespołów.
- *Budowa* — nowoczesne stocznie są bardzo skomputeryzowane. Proces składania okrętu niezwykle starannie planuje się przy użyciu komputera, a następnie wykorzystuje się komputer do kontrolowania prac. Spawaniem zajmują się roboty. W szczególności produkcja dwukadłubowych tankowców byłaby niemożliwa bez małych robotów spawających, które pracują w przestrzeni między kadłubami. Tam po prostu jest za mało miejsca dla człowieka. Jednym z pierwszych na świecie zastosowań technologii CAD/CAM (projektowanie i produkcja wspomagane komputerowo) było cięcie arkuszy stali na potrzeby budowy statków.
- *Silnik* — silnik ze swoim elektronicznym systemem wtrysku paliwa jest kontrolowany przez kilkadziesiąt komputerów. W przypadku silnika o mocy stu tysięcy koni mechanicznych (jak ten na fotografii) nie jest to łatwe zadanie. Na przykład komputery te bez przerwy dostosowują skład mieszanki paliwowej, aby zminimalizować ilość zanieczyszczeń, które wydostawałyby się ze źle dostrojonego silnika. Wiele pomp, z których korzysta silnik (i inne części statku), także jest skomputeryzowanych.
- *Zarządzanie* — statki wyruszają w rejs, gdy trzeba przetransportować ładunek. Planowanie rejsów flot statków jest procesem ciągłym (oczywiście skomputeryzowanym), którego celem jest dostosowanie się do zmian pogody, podaży i popytu oraz ilości miejsca i przepustowości portów. Istnieją nawet strony internetowe, na których można sprawdzić,

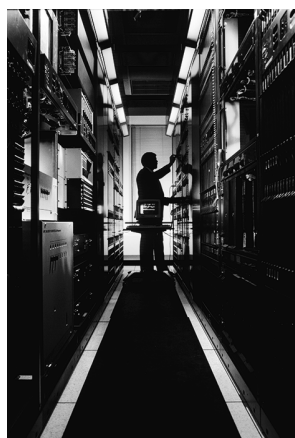
gdzie aktualnie znajdują się największe jednostki pływające. Okręt widoczny na fotografii jest kontenerowcem (jednym z największych na świecie — ma długość 397 metrów i szerokość 56 metrów), ale statkami innych rodzajów zarządza się w podobny sposób.

- *Monitorowanie* — statek, który wyrusza w rejsy po oceanie, jest w znacznym stopniu jednostką autonomiczną. To znaczy, że znajdująca się na jego pokładzie załoga może poradzić sobie z większością problemów, które mogą wystąpić przed dotarciem do portu. Statki należą też do globalnej sieci. Załoga ma dostęp do precyzyjnych prognoz pogody (które odbiera od skomputeryzowanych satelitów i za ich pośrednictwem). Jednostki są wyposażone w systemy GPS oraz komputerowo sterowane radary. Jeśli załoga musi udać się na spoczynek, większość systemów (włącznie z silnikiem, radarami itp.) można monitorować (przez satelitę) ze sterowni linii żeglugowej. Jeśli zostanie zauważone coś niezwykłego lub wystąpi przerwa w łączności, załoga zostanie o tym powiadomiona.

Pomyśl, jakie skutki mogłaby wywołać awaria jednego z setek wymienionych przeze mnie komputerów, nie mówiąc już o tych, których nawet nie wspominałem. Nieco bardziej szczegółowo temat ten zostanie opisany w rozdziale 25. („Programowanie systemów wbudowanych”). Pisanie kodu do obsługi nowoczesnego okrętu jest interesującym zadaniem, które wymaga wysokich kwalifikacji. Jest to także bardzo pożyteczna praca. Koszty transportu są naprawdę zaskakująco niskie. Korzystasz z tego, gdy kupujesz produkt, który nie został wyprodukowany tam, gdzie mieszkasz. Transport morski zawsze był tańszy od lądowego. Obecnie jednym z powodów, dlaczego tak jest, są komputery i informacja.

1.5.3. Telekomunikacja

Na poniższych fotografiach widać centralę telefoniczną i telefon (który przy okazji jest jeszcze aparatem fotograficznym, odtwarzaczem MP3, radiem FM, przeglądarką internetową i wieloma innymi rzeczami):



Zobaczmy, gdzie w tych przypadkach komputery i oprogramowanie odgrywają kluczową rolę. Podnosisz telefon, wybierasz numer i osoba, do której dzwonisz, odpowiada. Może się też zdarzyć, że porozmawiasz z pocztą głosową, zechcesz wysłać komuś zdjęcie zrobione aparatem cyfrowym wbudowanym w telefon albo wyślesz SMS-a (naciskasz tylko klawisz *Wyslij*, a resztą

zajmuje się już telefon). Telefon jest oczywiście komputerem. Jest to tym bardziej oczywiste, ponieważ większość telefonów komórkowych ma ekran i pozwala na wykonywanie wielu więcej czynności niż „stare tradycyjne aparaty”, np. przeglądanie stron internetowych. W istocie takie telefony najczęściej zawierają kilka komputerów — jeden do obsługi ekranu, jeden do rozmów i może jeszcze kilka do innych celów.

Ta część telefonu, która odpowiada za obsługę ekranu, surfowanie po internecie itp., jest prawdopodobnie najlepiej znana użytkownikom komputerów — obsługuje graficzny interfejs „wszystkich typowych funkcji”. To, o czym nie wie, a nawet nie podejrzewa większość użytkowników telefonów, to bardzo skomplikowany system, którym telefon posługuje się, aby wykonać swoje działania. Wybieram numer, będąc w Warszawie, a Ty, podczas spędzania wakacji w Kawkach, po kilku sekundach podnosisz słuchawkę i mówisz: „Halo!”, przekrzykując wycie krów na pastwisku. Większość telefonów można wykorzystać do połączenia dwóch osób znajdujących się w dwóch dowolnych miejscach na świecie i nikogo to nie dziwi. Jak mój telefon znalazł Twój? W jaki sposób został przesłany dźwięk? Jak odbywa się kodowanie dźwięku w pakiety danych? Odpowiedź na te pytania wymagałaby napisania wielu książek, o wiele grubszych od tej. Jedno jest pewne — w to wszystko zaangażowane są setki komputerów z oprogramowaniem porozrzucanych na obszarze, o którym mowa. Jeśli masz pecha, w proces ten zostanie zaangażowanych kilka satelitów komunikacyjnych (które też są skomputeryzowanymi systemami) — jest to pech, ponieważ nie da się zrekompensować opóźnienia podróży na odległość ponad 32 tysięcy kilometrów w przestrzeni kosmicznej. Prędkość światła (a więc i Twojego głosu) nie jest nieskończona (znacznie lepsze są światłowody — krótsze, szybsze i pozwalają na przesył znacznie większej ilości danych). Większość tych systemów działa nadzwyczaj dobrze. Szkieletowe systemy telekomunikacyjne cechuje niezawodność na poziomie 99,9999% (a więc nie działają średnio przez 20 minut na 20 lat — tj. 20/20·365·24·60). Problemy, których doświadczamy, mają najczęściej związek z połączeniem telefonu z najbliższą centralą telefoniczną.

Istnieje oprogramowanie łączące nasze telefony, dzielące wypowiedziane przez nas słowa na pakiety danych, które są następnie wysyłane przewodami i drogą radiową, wyznaczające trasę tych pakietów, pomagające naprawić wszelkiego rodzaju awarie, bez przerwy monitorujące jakość i niezawodność świadczonych usług i oczywiście naliczające wysokość opłat. Nawet zaplanowanie nad fizycznymi elementami systemu wymaga dużych ilości sprytnie zaprojektowanego oprogramowania. Co z czym się kontaktuje? Jakie elementy znajdują się w nowym systemie? Kiedy trzeba wykonać profilaktyczne prace konserwacyjne?

Zapewne światowy szkieletowy system telekomunikacyjny, który składa się z częściowo niezależnych i wzajemnie powiązanych mniejszych systemów, jest największym i najbardziej skomplikowanym wytworem w historii człowieka. Aby było jasne: to nie jest tylko stara dobra telefonia z kilkoma dodatkami. Scaleniu uległy różne infrastruktury. To one podtrzymują internet, umożliwiają działanie systemów bankowych i handlowych oraz pośredniczą w przysyłaniu programów telewizyjnych do stacji nadawczych. Na następnej stronie znajdują się jeszcze dwie fotografie przedstawiające telekomunikację.

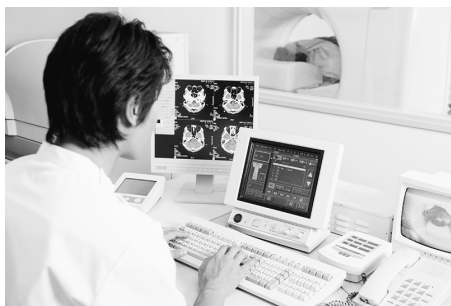
Po lewej widać „parkiet” amerykańskiej giełdy na Wall Street w Nowym Jorku. Po prawej znajduje się mapa sieci szkieletowych internetu (gdyby przedstawiono wszystkie, nic by nie dało się z tego rysunku wynioskować).

Jak wiadomo, wszyscy lubią też robić zdjęcia aparatem cyfrowym i wykorzystywać komputery do rysowania specjalnych map przedstawiających graficznie pewien stan wiedzy.



1.5.4. Medycyna

Na poniższych fotografiach widać skaner do wykonywania tomografii komputerowej i salę operacyjną, na której jest przeprowadzana operacja przy wykorzystaniu komputerów (czasami technika ta nazywana jest chirurgią wspomaganą robotami):



Zastanówmy się, jaka jest rola komputerów w tych przypadkach. Skanery są w istocie komputerami. Wysyłane przez nie sygnały są kontrolowane przez komputer. Ponadto dane, które zwracają, nie nadają się do użytku, dopóki nie zostaną przetworzone na rozpoznawalne dla człowieka trójwymiarowe obrazy części ludzkiego ciała przez specjalnie zaprojektowane do tego celu wyrafinowane algorytmy. Chirurgia wspomagana komputerowo wymaga jeszcze bardziej zaawansowanych technik. Dzięki różnym technikom wizualizowania chirurg może obejrzeć wnętrze pacjenta, znacznie powiększyć obraz operowanego miejsca lub je lepiej oświetlić, niż byłoby to możliwe przy zastosowaniu tradycyjnych technik. Dzięki pomocy komputerów chirurg może używać narzędzi, które są zbyt wyrafinowane, aby mogła nimi manipulować ludzka ręka, i dotrzeć tam, gdzie bez tych narzędzi konieczne byłoby wykonywanie nacięć. Chirurgia polegająca na minimalnej inwazyjności (laparoscopia) jest prostym przykładem tego, jak nowe technologie przyczyniły się do zmniejszenia cierpienia i skrócenia czasu powrotu do pełni zdrowia milionów ludzi na całym świecie. Komputery mogą także pomóc w opanowaniu „ręki” chirurga, dzięki czemu możliwe jest wykonanie delikatniejszych prac, które bez tego byłyby niewykonalne. W końcu „robotem” można sterować na odległość, dzięki czemu doktor może pomagać chorym na odległość (przez internet). Wykorzystywane w medycynie komputery i oprogramowanie są zadziwiające, skomplikowane i bardzo ciekawe. Same interfejsy użytkownika, sterowanie urządzeniami i techniki obrazowania zapewnią mnóstwo pracy tysiącom naukowców, inżynierów i programistów na długie lata.

Duża grupa lekarzy prowadziła dyskusję na temat tego, które z nowych narzędzi jest im najbardziej przydatne. Wymieniano np. skanery do tomografii komputerowej, skanery MRI, urządzenia automatycznie wykonujące badanie krwi, wysokiej rozdzielczości urządzenia ultradźwiękowe i komputery kieszonkowe. Wynik dyskusji był zaskakujący, ponieważ zwycięzcą okazał się natychmiastowy dostęp do danych pacjenta. Znajomość karty medycznej pacjenta (przebyte choroby, przyjmowane wcześniej leki, alergię, choroby dziedziczne, ogólny stan zdrowia, bieżąca kuracja itp.) ułatwia diagnozę i minimalizuje ryzyko popełnienia błędu.

1.5.5. Informacja

Na poniższych ilustracjach widać typowy komputer PC (a nawet dwa) i fragment serwerowni:



Skoncentrowaliśmy się na gadżetach z oczywistego powodu — oprogramowania nie da się zobaczyć ani usłyszeć. Nie możemy pokazać zdjęcia fajnego programu, ale możemy pokazać urządzenie, na którym on działa. Zadaniem wielu programów jest jednak bezpośrednio przetwarzanie danych. Przyjrzymy się teraz „zwykłemu sposobowi użycia zwykłych komputerów”, na których działa „zwykłe oprogramowanie”.

Serwerownia to zespół komputerów dostarczających usługi sieciowe. Organizacje posiadające najnowocześniejsze serwerownie (takie jak Google, Amazon i Microsoft) bardzo niechętnie dzielą się informacjami na temat swoich serwerów, poza tym parametry ich infrastruktury cały czas się zmieniają (przez co większość informacji, jakie znajdziesz w internecie, będzie nieaktualna). Niemniej jednak specyfikacje robią wrażenie i powinny przekonać do zmiany zdania każdego, kto myśli, że programowanie polega tylko na wykonaniu paru obliczeń na laptopie:

- Google ma około miliona serwerów (każdy mocniejszy od Twojego laptopa) rozmieszczonych w 25 – 50 centrach danych.
- Takie centrum danych może mieścić się w budynku o wymiarach nawet 60×100 m i więcej.
- W 2011 r. gazeta „New York Times” napisała, że centra danych Google’a nieustannie pobierają około 260 milionów watów energii (mniej więcej tyle co Las Vegas).
- Powiedzmy, że typowy serwer ma czterordzeniowy procesor z zegarem o częstotliwości 3 GHz i 24 GB pamięci głównej. To oznacza, że dysponuje mocą obliczeniową wynoszącą około $12 \cdot 10^{15}$ Hz (około 12 000 000 000 000 000 instrukcji na sekundę) i $24 \cdot 10^{15}$ bajtów pamięci głównej (około 24 000 000 000 000 000 8-bitowych bajtów) oraz jakimś 4 TB, czyli $4 \cdot 10^{18}$, przestrzeni dyskowej.

Możliwe, że te dane są i tak zaniżone, a zanim ta książka dotrze w Twoje ręce, prawie na pewno będą. W szczególności ostatnimi czasy inżynierowie starają się ograniczyć zużycie energii, stosując więcej procesorów na serwer i wbudowując więcej rdzeni w procesor. Jednostka GB oznacza gigabajt informacji, tj. około 1 000 000 000 znaków. Terabajt (TB) to około 1000 GB, tj. około 1 000 000 000 000 znaków. Obecnie serwerownie są znacznie większe. Jest to ekstremalny przykład, ale wszystkie duże firmy mają programy działające w sieci, które pośredniczą w jej komunikacji z użytkownikami i klientami. Jako przykłady takich firm można wymienić Amazon (sklep internetowy), Amadeus (sprzedaż biletów lotniczych i wynajem samochodów) oraz Ebay (aukcje internetowe). W sieci obecne są także miliony małych firm, organizacji i osób prywatnych. Większość z nich nie korzysta z własnego oprogramowania, ale duża ich część go używa i nie jest ono banalne.

Bardziej tradycyjny sposób wykorzystania komputerów to prowadzenie księgowości, przetwarzanie zamówień, obsługa list płac, archiwizacja danych, pobieranie opłat, zarządzanie magazynem, zarządzanie danymi pracowników, studentów, pacjentów itd. — danymi, które w zasadzie każda organizacja (komercyjna i niekomercyjna, rządowa i pozarządowa) przechowuje. Dane te są podstawą działalności tych organizacji. Jeśli chodzi o rolę komputerów, przetwarzanie takich danych wydaje się proste. Część z nich jest głównie przechowywana i od czasu do czasu sprawdzana, bez wielkich modyfikacji. Na przykład:

- Czy mój lot o 12:30 do Chicago nie opóźni się?
- Czy Gilbert Sullivan chorował na świnkę?
- Czy ekspres do kawy zamówiony przez Waldemara Flaka został już wysłany?
- Jakiego rodzaju krzesła kuchenne kupił w 1996 roku Jan Mogiła?
- Ile rozmów telefonicznych zainicjowano w sierpniu 2012 roku w województwie mazowieckim?
- Ile sprzedano filiżanek do kawy w styczniu i za jaką łączną sumę pieniędzy?

Sama wielkość baz danych sprawia, że systemy informatyczne są bardzo skomplikowane. Do tego należy dodać potrzebę szybkiego zwracania odpowiedzi (często musi to być czas nie dłuższy niż dwie sekundy dla pojedynczego zapytania) oraz potrzebę poprawności (przynajmniej w większości przypadków). Obecnie często posługujemy się jednostką terabajt, rozmawiając o danych (jeden bajt to ilość pamięci potrzebna do zapisania jednego znaku). Jest to tradycyjny sposób przetwarzania danych, który scala się z siecią, ponieważ w większości przypadków dostęp do baz danych odbywa się przez interfejsy sieciowe.

Ten sposób wykorzystania komputerów często nazywa się **przetwarzaniem informacji**. W centrum zainteresowania są dane — zazwyczaj duże ich ilości. To tworzy wyzwania dotyczące organizacji i przesyłu danych oraz wymaga opracowania sposobów przedstawiania ich olbrzymich ilości w zrozumiały sposób. Ważnym aspektem obsługi danych są interfejsy użytkownika. Pomyśl na przykład o analizie klasycznych dzieł literackich (np. *Ogniem i mieczem* Sienkiewicza lub *Straconych złudzeń* Balzaca), których celem jest odkrycie, co autor naprawdę miał na myśli, poprzez porównywanie wielu wersji. Musimy przeszukać tekst przy zastosowaniu wielu różnych kryteriów dostarczonych przez badacza tekstu oraz wyświetlić wyniki w taki sposób, aby mógł on odkryć istotne rzeczy. Kiedy mowa o analizie tekstu, na myśl nasuwa się

publikowanie artykułów, książek, broszur, gazet itp. Wszystkie są opracowywane na komputerze. Oprogramowanie, które w pełni spełniałoby wymagania w tym zakresie, dla wielu ludzi nadal pozostaje w sferze marzeń.

1.5.6. Sięgamy w kosmos

Niektórzy twierdzą, że paleontolog może na podstawie jednej małej kości dinozaura zrekonstruować całe zwierzę oraz opisać środowisko, w którym ono żyło, i styl jego życia. To lekka przesada, ale bez wątpienia można domyślić się pewnych faktów, sądząc po wyglądzie i kształcie przedmiotu. Spójrz na zdjęcie powierzchni Marsa zrobione aparatem umieszczonym na jednym z łazików wysłanych tam przez NASA:



Jeśli chcesz być badaczem kosmosu, dobrym sposobem na dostanie takiej pracy jest bycie dobrym programistą. W rozmaitych programach badawczych przestrzeni kosmicznej (z udziałem człowieka w kosmosie i bez) zatrudnionych jest mnóstwo projektantów oprogramowania, zwłaszcza takich, którzy dodatkowo znają przynajmniej podstawy fizyki, matematyki, elektrotechniki, mechaniki, technologii medycznych itp. Sprawienie, że te dwa łaziki poruszały się po powierzchni Marsa przez lata jest jednym z najwspanialszych osiągnięć technologicznych naszej cywilizacji. Jeden z nich (Spirit) wysyłał dane przez sześć lat, a drugi (Opportunity) — wciąż działał podczas pisania tej książki i w styczniu 2014 r. obchodził dziesiąty jubileusz pobytu na Marsie. A planowano, że urządzenia te będą działać tylko trzy miesiące.

To zdjęcie dotarło do Ziemi kanałem komunikacyjnym z 25-minutowym opóźnieniem. Aby zminimalizować liczbę przesyłanych bitów bez utraty nawet jednego, trzeba było opracować zaawansowane algorytmy przy wykorzystaniu wyrafinowanych metod matematycznych. Na Ziemi zdjęcie zostało odtworzone przy użyciu algorytmów odzyskujących kolory i minimalizujących zniekształcenia powodowane przez elektroniczne i optyczne czujniki.

Do kontroli łazików na Marsie wykorzystuje się oczywiście programy. Poruszają się one autonomicznie, wykonując polecenia, które zostały do nich wysłane z Ziemi poprzedniego dnia. Transmisję sygnałów obsługują programy.

Systemy operacyjne sterujące różnymi komputerami związanymi z łazikami, transmisją danych i rekonstrukcją zdjęć są tak samo programami jak te, których używam do pisania tego tekstu. Komputery, na których działają, zostały zaprojektowane przy użyciu narzędzi CAD/CAM. Znajdujące się w tych komputerach układy scalone zostały wyprodukowane na skomputeryzowanych liniach produkcyjnych, które skonstruowano przy użyciu precyzyjnych narzędzi. Te narzędzia z kolei także są projektowane i produkowane przy użyciu komputerów i oprogramowania. Kontrola jakości tych długich procesów produkcji także wymaga dużego nakładu

pracy komputerów. Cały kod został napisany przez ludzi w języku programowania wysokiego poziomu i przetłumaczony przez kompilatory, które same są programami, na język maszynowy. Wiele z tych programów porozumiewa się z użytkownikiem za pośrednictwem graficznych interfejsów oraz strumieni wejścia i wyjścia.

W końcu wiele programów tworzy się do obróbki grafiki (także zdjęć dostarczanych przez łaziki z Marsa), tworzenia animacji oraz edycji zdjęć (w sieci można znaleźć zdjęcia z łazików, na których widać obcych).

1.5.7. I co z tego



Co te wszystkie wymyślne i skomplikowane programy i systemy oprogramowania mają wspólnego z nauką programowania i używaniem języka C++? Związek jest taki, że wielu programistów pracuje nad takimi właśnie projektami. Są to tego rodzaju rzeczy, w których bardzo pomocne są dobre programy. Ponadto w każdym z przedstawionych do tej pory przykładów zostały wykorzystane język C++ oraz część technik, które opisujemy w tej książce. Tak, programy napisane w języku C++ można znaleźć w odtwarzaczach MP3, na okrętach, w turbinach wiatrowych, na Marsie i w projekcie badania ludzkiego genomu. Więcej zastosowań tego języka opisano na stronie www.stroustrup.com/applications.html.

1.6. Ideały dla programistów



Do czego mają służyć nasze programy? Czego chcemy ogólnie, nie mówiąc o poszczególnych funkcjach konkretnego programu? Chcemy, aby programy były **poprawne** i w związku z tym **niezawodne**. Jeśli program nie działa zgodnie z oczekiwaniami, ale działa na tyle dobrze, że da się z niego korzystać, to w najlepszym przypadku będzie denerwujący, a w najgorszym niebezpieczny. Chcemy, aby był **dobrze zaprojektowany** i dobrze spełniał swoje zadanie. Nie ma w istocie znaczenia, czy program jest poprawny, jeśli to, co robi, jest nam nieprzydatne, lub jeśli robi coś poprawnie, ale w denerwujący sposób. Chcemy też, aby jego cena była **przystępna**. Może wolałbym kupić sobie Rolls-Royce'a albo osobisty odrzutowiec, ale jeśli nie jestem miliarderem, wielki wpływ na mój wybór środka transportu będzie miała cena.



To są te aspekty oprogramowania (gadżetów i systemów), którymi mogą cieszyć się użytkownicy, nie programiści. Do tych ideałów programiści powinni dążyć. Muszą o nich pamiętać cały czas, zwłaszcza w początkowych fazach pracy, jeśli chcą, aby ich oprogramowanie odnosiło sukcesy. Ponadto musimy pamiętać o ideałach związanych z samym kodem, który musi być łatwy w **utrzymaniu**, tzn. ktoś, kto go nie napisał nie powinien mieć problemów z jego zrozumieniem i modyfikowaniem. Udany program „żyje” przez wiele lat (często nawet dekad) i jest w tym czasie wielokrotnie modyfikowany. Zostanie przystosowany do nowej platformy sprzętowej, wzbogaci się o nowe funkcje, zostanie dodana obsługa nowych urządzeń wejściowych (ekranów, odtwarzaczy video i dźwięku), zostaną dodane nowe moduły językowe itp. Tylko nieudanych programów nigdy się nie modyfikuje. Aby program był łatwy w utrzymaniu, musi być względnie prosty, biorąc pod uwagę jego przeznaczenie, a jego kod źródłowy musi bezpośrednio reprezentować wyrażone w nim myśli. Złożoność — wróg prostoty i łatwości utrzymania — może być nie do uniknięcia w niektórych przypadkach (wówczas trzeba sobie po prostu z nią poradzić). Może jednak też powstać, jeśli ktoś źle wyrazi swoje myśli za pomocą kodu. Należy tego unikać, stosując dobry styl programowania — styl jest ważny!

To nie wydaje się bardzo trudne, ale jest. Dlaczego? Podstawowe założenie programowania jest proste: powiedzieć maszynie, co ma robić. Dlaczego więc programowanie potrafi być tak trudne? Zasadniczo komputery to proste urządzenia. Potrafią wykonać tylko kilka operacji, np. zsumować dwie liczby i dokonać wyboru następnej instrukcji do wykonania na podstawie wyniku porównania dwóch liczb. Problem w tym, że my chcemy, aby komputery robiły skomplikowane rzeczy. Chcemy, aby robiły to, co jest dla nas za trudne, ale one są marudnymi, mściwymi i głupimi stworzeniami. Co więcej, świat jest znacznie bardziej skomplikowany, niż byśmy chcieli, przez co nie wiemy, jakie będą rzeczywiste implikacje spełnienia naszych próśb. Chcemy po prostu, aby program zrobił „coś takiego” bez zagłębiania się w techniczne szczegóły. Często też korzystamy ze „zdrowego rozsądku”. Niestety zdrowy rozsądek nie jest cechą wszystkich ludzi, a komputery są go w ogóle pozbawione (choć niektóre dobrze zaprojektowane programy potrafią go w pewnych dobrze zrozumianych przypadkach imitować).

Taki sposób myślenia prowadzi do wniosku, że „programowanie oznacza zrozumienie” — jeśli potrafisz zaprogramować rozwiązanie jakiegoś problemu, znaczy, że go rozumiesz. I odwrotnie, jeśli dobrze rozumiesz jakiś problem, możesz napisać program, który go rozwiąże. Innymi słowy programowanie można postrzegać jako próbę dogłębnego zrozumienia problemu.

Programując, znaczną część czasu spędza się na próbach zrozumienia natury zadania, które chce się zautomatyzować.

W procesie tworzenia programu można wyróżnić cztery etapy:

- *Analiza.* Na czym polega problem? Czego chce użytkownik? Czego potrzebuje użytkownik? Na co użytkownik może sobie pozwolić? Które elementy programu bezwzględnie muszą być niezawodne?
- *Projektowanie.* Jak rozwiążemy problem? Jak powinna wyglądać ogólna struktura całego systemu? Z jakich części będzie się składał? W jaki sposób części te będą się ze sobą komunikować? Jak system będzie komunikował się z użytkownikami?
- *Programowanie.* Wyrażenie rozwiązania problemu w postaci kodu. Pisanie kodu z uwzględnieniem wszystkich czynników ograniczających (czas, przestrzeń, pieniądze, niezawodność itp.). Kod powinien być poprawny i łatwy w utrzymaniu.
- *Testowanie.* Zapewnij prawidłowe działanie programu we wszystkich wymaganych warunkach, systematycznie przeprowadzając testy.

Programowanie w połączeniu z testowaniem często określa się mianem **implementacji**. Oczywiście przedstawiony powyżej podział procesu tworzenia oprogramowania na cztery etapy jest dużym uproszczeniem. Na temat każdego z nich napisano wiele grubych książek i jeszcze grubsze na temat tego, co je łączy. Należy pamiętać, że etapy te nie są niezależne i nie następują ściśle jeden po drugim. Najczęściej pierwszym etapem jest analiza, ale informacje uzyskane dzięki testom mogą pomóc w usprawnieniu programowania. Problemy związane ze zmuszeniem programu do działania mogą wskazywać na wady projektu, a w trakcie pracy nad projektem może się okazać, że pewne aspekty problemu mają swoje źródło jeszcze w fazie analizy. W istocie podczas użytkowania systemu wychodzą na jaw słabości analizy.

Największe znaczenie ma w takim przypadku **informacja zwrotna**. Uczymy się na własnych błędach i zmieniamy swoje postępowanie na podstawie wyciągniętych z lekcji wniosków. Ma to kluczowe znaczenie dla efektywności programowania. W żadnym dużym projekcie nie ma jednej osoby, która wie wszystko na temat problemu i jego rozwiązania od samego początku.

Możemy wypróbowywać różne pomysły i uzyskiwać informacje, pisząc kod, ale we wczesnych fazach pracy łatwiej jest (i szybciej) zdobyć potrzebne informacje, zapisując pomysły na temat projektowania i wypróbowując je na przyjaciółach przy zastosowaniu różnych scenariuszy wykorzystania. Najlepszym znanym nam narzędziem jest zielona tablica (jeśli wolisz zapach środków chemicznych od kurzu kredowego, możesz użyć białej tablicy). Nigdy nie przystępuj do projektowania w pojedynkę, jeśli nie musisz! Nie zaczynaj pisanie kodu, dopóki nie zweryfikujesz swoich pomysłów w drodze konsultacji z kimś innym. Zanim ruszysz w kierunku klawiatury, omów swoje techniki programowania i projektowania ze znajomymi, potencjalnymi użytkownikami itp. Zadziwiające, jak dużo można się nauczyć, tylko próbując wyrazić za pomocą słów swoje pomysły. W końcu program to nic innego jak wyrażenie za pomocą kodu myśli programisty.

Analogicznie, jeśli utkniesz w martwym punkcie, implementując program, oderwij wzrok od klawiatury i pomyśl o samym problemie, a nie swoim niekompletnym rozwiązaniu. Porozmawiaj z kimś, wyjaśnij, co chcesz zrobić i czemu to nie działa. Zadziwiające, jak często udaje się odkryć rozwiązanie problemu podczas opisywania go komuś innemu. Nie zajmuj się debugowaniem (znajdowaniem błędów w kodzie) w pojedynkę, jeśli nie musisz!

Głównym tematem tej książki jest implementowanie, a zwłaszcza programowanie. Nie uczymy rozwiązywania problemów, poza tym że przedstawiamy mnóstwo przykładów problemów wraz z rozwiązaniami. Rozwiązywanie problemów w dużej mierze polega na rozpoznaniu problemu i rozwiązaniu go za pomocą znanej techniki. Tylko gdy większość podproblemów jest rozwiązywana w ten sposób, można pozwolić sobie na ekscytujące i kreatywne myślenie prowadzące do nieszablonowych rozwiązań. Zatem koncentrujemy się na nauce jasnego wyrażania myśli za pomocą kodu.



Bezpośrednie wyrażanie myśli w kodzie jest podstawowym ideałem programisty. To jest oczywiste, ale jak do tej pory nie zostały przedstawione żadne popierające tę tezę przykłady. Będziemy do tego wracać wielokrotnie. Jeśli potrzebujemy w programie liczby całkowitej, zapisujemy ją w zmiennej typu `int`, która umożliwia wykonywanie podstawowych działań na tego typu liczbach. Jeśli potrzebujemy łańcucha znaków, używamy zmiennej typu `string`, która umożliwia wykonywanie najbardziej podstawowych operacji na tekście. Najbardziej podstawową zasadą jest, że jeśli mamy jakiś pomysł, element, jakieś pojęcie, coś, o czym możemy pomyśleć „rzecz”, coś, co możemy narysować na tablicy i omówić z innymi, o czym wspomniano w książce (nie o komputerach), wówczas chcemy, aby to coś w naszym programie miało nazwę (typ) oraz pozwalało na wykonywanie operacji, które naszym zdaniem są odpowiednie. Jeśli chcemy zajmować się matematyką, potrzebujemy typów: `Complex`, reprezentującego liczby zespolone, i `Matrix`, reprezentującego macierze. Do grafiki potrzebujemy typów: `Shape`, `Circle`, `Color` oraz `Dialog_box`. Jeśli planujemy odbierać dane wejściowe, np. z czujnika temperatury, wykorzystamy typ `istream` (i oznacza *input*, czyli dane wejściowe). Oczywiście każdy z wymienionych typów powinien umożliwiać wykonywanie odpowiednich operacji i tylko takich. To tylko kilka przykładów z tej książki. Poza tym przedstawimy Ci narzędzia i techniki, za pomocą których będziesz tworzyć własne typy bezpośrednio reprezentujące wszystko, czego zechcesz użyć w swoim programie.

Programowanie to zajęcie częściowo praktyczne, a częściowo teoretyczne. Jeśli skupisz się tylko na pierwszym aspekcie, będziesz tworzyć nieskalowalne i trudne w utrzymaniu gnioty. W drugim przypadku będą powstawać bezużyteczne (lub zbyt drogie) zabawki.

Inne sposoby postrzegania ideałów programistycznych i kilka osób, które w dużym stopniu zasłużyły się tej dziedzinie, wykorzystując w pracy języki programowania, zostały opisane w rozdziale 22., „Ideały i historia”.

Powtórzenie

Pytania powtórzeniowe mają na celu zwrócić Twoją uwagę na najważniejsze pojęcia opisane w rozdziale. W pewnym sensie stanowią uzupełnienie ćwiczeń, które koncentrują się na praktycznych aspektach programowania. Natomiast pytania powtórzeniowe mają za zadanie pomóc Ci wyrazić na głos swoje myśli. W tym sensie przypominają dobrze dobrane pytania na rozmowie kwalifikacyjnej.

- 1. Co to jest oprogramowanie?
- 2. Dlaczego oprogramowanie jest ważne?
- 3. W jakich dziedzinach oprogramowanie ma znaczenie?
- 4. Co może się stać, jeśli jakiś program źle zadziała? Podaj kilka przykładów.
- 5. W jakich sferach oprogramowanie odgrywa ważną rolę? Podaj kilka przykładów.
- 6. Wymień kilka zawodów związanych z produkcją oprogramowania.
- 7. Jaka jest różnica między informatyką a programowaniem?
- 8. Gdzie w projektowaniu, konstruowaniu i użytkowaniu okrętów jest wykorzystywane oprogramowanie?
- 9. Co to jest serwerownia?
- 10. Jakiego rodzaju pytania najczęściej zadaje się przez internet? Podaj kilka przykładów.
- 11. Wymień kilka zastosowań oprogramowania w nauce.
- 12. Wymień kilka zastosowań oprogramowania w medycynie.
- 13. Wymień kilka zastosowań oprogramowania w przemyśle rozrywkowym.
- 14. Jakie ogólne cechy powinno mieć dobre oprogramowanie?
- 15. Jak wygląda typowy programista?
- 16. Wymień etapy procesu produkcji oprogramowania.
- 17. Wymień kilka czynników powodujących, że programowanie bywa trudnym zajęciem.
- 18. Wymień kilka zastosowań oprogramowania, które ułatwiają życie.
- 19. Wymień kilka zastosowań oprogramowania, które utrudniają życie.

Terminologia

W tej sekcji przedstawiamy podstawowe słownictwo związane ogólnie z programowaniem i językiem C++. Jeśli chcesz rozumieć ludzi rozmawiających na tematy programistyczne i móc włączać się do tych dyskusji, powinieneś znać to słownictwo.

analiza	klient	projekt
CAD/CAM	komunikacja	przystępność
GUI	oprogramowanie	stereotyp
ideały	poprawność	testowanie
implementacja	programista	użytkownik
informacje zwrotne	programowanie	zielona tablica

Praca domowa

1. Pomyśl o jakiejś czynności, którą wykonujesz każdego dnia (np. udział w zajęciach, jedzenie obiadu albo oglądanie telewizji). Sporządź listę sposobów, w jakie komputery pośrednio lub bezpośrednio są zaangażowane w te czynności.
2. Pomyśl o jakimś zawodzie, najlepiej takim, który przynajmniej trochę znasz. Sporządź listę czynności, do których ludzie go wykonujący wykorzystują komputery.
3. Wymień się z kolegą lub koleżanką listą z zadania drugiego i dodaj własne przemyślenia do cudzej listy. Porównajcie swoje wyniki. Pamiętaj: takie otwarte ćwiczenia nie mają jednego dobrego rozwiązania, zawsze można coś zmienić.
4. Opisz jakąś czynność z własnego doświadczenia, której wykonanie bez komputerów nie byłoby możliwe.
5. Sporządź listę programów, z których bezpośrednio korzystasz. Uwzględnij tylko te przypadki, w których interakcja z programem jest sprawą oczywistą (np. wybór kolejnego utworu w odtwarzaczu MP3). Nie licz wątpliwych przypadków, jak skręcanie koła kierownicy w samochodzie.
6. Sporządź listę dziesięciu czynności, do których wykonania ludzie nie wykorzystują w żaden sposób komputerów. To może być trudniejsze, niż myślisz!
7. Wymień pięć czynności, które nie są obecnie wykonywane przy użyciu komputerów, ale Twoim zdaniem będą w przyszłości. Napisz kilka zdań na temat każdej z nich.
8. Napisz krótki tekst (od 100 do 500 słów), w którym wyjaśnisz, dlaczego chcesz zostać programistą. Jeśli jesteś przekonany, że nie chcesz nim zostać, wyjaśnij to. Bez względu na decyzję postaraj się zaprezentować dobrze przemyślaną i logiczną argumentację.
9. Napisz krótki tekst (od 100 do 500 słów), w którym wyjaśnisz, jaką rolę, poza rolą programisty, chciałbyś pełnić w informatyce (bez względu na to, czy Twoim głównym zainteresowaniem jest bycie programistą).
10. Czy uważasz, że komputery kiedykolwiek staną się świadomie myślącymi istotami mogącymi konkurować z ludźmi? Napisz krótki tekst (nie mniej niż 100 słów), w którym wyjaśnisz swoje stanowisko.
11. Wymień kilka cech dobrego programisty. Następnie wymień kilka cech, które są najczęściej przypisywane programistom.
12. Wymień przynajmniej pięć rodzajów dziedzin omówionych w tym rozdziale, w których wykorzystywane są komputery. Wybierz jedną, która wydaje Ci się najbardziej interesująca i chciałbyś kiedyś wziąć udział w związanym z nią projekcie. Swoją wybiórcę opisz w krótkim wypracowaniu o długości przynajmniej 100 słów.
13. Ile pamięci trzeba na zapisanie: a) tekstu z tej strony; b) tekstu z tego rozdziału; c) wszystkich dzieł Shakespeare'a? Przyjmij założenie, że każdy znak zajmuje jeden bajt, oraz spróbuj zgadnąć z dokładnością do 20%.
14. Jaką ilość pamięci ma Twój komputer? Ile ma pamięci RAM, a jaką pojemność ma dysk twardy?

Podsumowanie

Nasza cywilizacja opiera się na oprogramowaniu. Jest ono nieprzebranym źródłem rozmaitych możliwości do znalezienia interesującej, przydatnej społecznie i dochodowej pracy. Jeśli bierzesz się za programowanie, bądź zdyscyplinowany i rób to poważnie. Masz wnieść wkład w rozwiązanie, a nie stać się częścią problemu.

Z podziwem obserwujemy, jak różnorodne rodzaje oprogramowania przenikają naszą wypełnioną technologią cywilizację. Oprogramowanie nie zawsze jest wykorzystywane do dobrych celów, ale to już inna historia. Tu chcemy pokazać, jak wszechobecne jest oprogramowanie oraz jak dużo od niego zależy każdego dnia. W całości zostało napisane przez takich ludzi, jak my. Wszyscy naukowcy, matematycy, inżynierowie, programiści itp., którzy zostali tu poruszeni, zaczęli w taki sam sposób, jak Ty teraz.

Czas zejść na ziemię i rozpocząć naukę technicznych umiejętności programistycznych. Jeśli zaczną nachodzić Cię myśli, czy to wszystko jest warte Twojej ciężkiej pracy (każdy myślący człowiek miewa takie chwile), przeczytaj jeszcze raz ten rozdział, Wstęp oraz fragmenty rozdziału 0 („Uwagi do czytelnika”). Jeśli najdą Cię wątpliwości, czy potrafisz stać się kompetentnym programistą, przypomnij sobie, że udało się to milionom programistów, projektantów, inżynierów oprogramowania itp. Tobie też może się udać.



