WPROWADZAJĄC… PORZĄDEK

Maciej M. Sysło
Wydział Matematyki i Informatyki
UMK w Toruniu, Uniwersytet Wrocławski
syslo@mat.umk.pl; syslo@ii.uni.wroc.pl, http://mmsyslo.pl

Abstract. This is a review of some important issues related to a new core computer science (informatics – Informatyka) curriculum and its implementation in schools in Poland. Porządek (order) refers here to at least two areas of interests: (1) historical background of continuous development of informatics education in Poland and (2) how to make students interested and engaged in computer science education through 12 years in schools by using just two concept – order and sorting – and related to them: problem situations, algorithms, programs etc.

1. Wstęp

Propozycja nowej podstawy programowej kształcenia informatycznego, czyli przedmiotu informatyka przez wszystkie lata w szkole, jawić się może jako rewolucja, a faktycznie jest ewolucją, jeśli spojrzy się wstecz na historię edukacji informatycznej w Polsce. Formalnie, za początek można przyjąć pierwszy program nauczania elementów informatyki, opracowany przez zespół PTI pod kierunkiem prof. Stanisława Waligórskiego, chociaż pierwsze regularne zajęcia informatyczne odbywały się już w połowie lat 60' XX wieku w liceach we Wrocławiu i w Warszawie. Od 1985 roku zadbaliśmy, by informatyka nie zniknęła z podstawy programowej, wbrew światowym tendencjom i decyzjom.

Proponowana zmiana ma jednak pewne cechy rewolucji – jej lejtmotywem jest **myślenie komputacyjne**, czyli, jak to niektórzy ujmują, myślenie, „jak myśli informatyk", a faktycznie posługiwanie się *mental tools* – metodami rozumowania i rozwiazywania problemów, które mają swoje źródła w informatyce – wiedzy o komputerach i ich wykorzystaniu. Cech rewolucyjnych można także upatrywać w propozycji kształtowania umiejętności programowania od najmłodszych lat, jednak z solidnym przygotowaniem uczniów wcześniej do tego dialogu z komputerem, przygotowania, by mieli o czym „rozmawiać" z komputerem.

Proponowana jest więc postępująca rewolucja, czy też ewolucyjna rewolucja, z którą nadążyć przyjdzie nauczycielom – i to jest kolejna zmiana rewolucyjna,

chcemy jednak, by nauczyciele przystąpili do ewolucyjnego rozwoju swojej wiedzy, umiejętności i kompetencji informatycznych – o tym jest mowa w innym moim tekście w tych materiałach, w którym przedstawiam standardy, jako wspólne odniesienie dla nauczycieli, starających się przygotować do uczenia informatyki „na nowo", jak i instytucji, które będą starały się swoja ofertą wspomóc w tym nauczycieli.

Dalej, podsumowuję historię edukacji informatycznej opisem dzisiejszego stanu, który napawa optymizmem w świetle czekających nas wyzwań. Następnie, przedstawiam wybrane elementy nowej podstawy i omawiam właśnie te wyzwania, których realizacja w nadchodzących latach zadecyduje o powodzeniu prezentowanej tutaj i na tej konferencji inicjatywy. Na końcu, ilustruję, jak znaczenie pojęć porządku i porządkowania oraz ich wykorzystanie może angażować uczniów przez 12 lat w szkole, nie tylko na zajęciach informatycznych.

2. Porządkowanie pola informatyki szkolnej

Propozycję nowej podstawy programowej, z całą filozofią, która jej towarzyszy (opisana w jej wstępie i w krążącej preambule, a wcześniej, zarysowanej w „Wizji" [6]), a, ku naszemu zadowoleniu, jest podejmowana przez wielu nauczycieli i edukatorów, można uznać za próbę uporządkowania roli komputerów, a ogólniej – technologii, i informatyki w edukacji. Nie miejsce tutaj, by prześledzić całą historię komputerów w edukacji i edukacji informatycznej w polskiej szkole, temu będzie poświęcone osobne opracowanie [8]. Warto jednak zauważyć i wypadałoby docenić, że obecna propozycja zmian w edukacji informatycznej jest naturalną konsekwencją rozwoju tej edukacji od jej początków w połowie lat 60' XX wieku. Na pytanie, czy polska szkoła jest przygotowana na proponowane zmiany odpowiadam i przekonuję, że tak. Spójrzmy więc na te argumenty, oddające aktualny stan:

1. **Wydzielone przedmioty informatyczne** są na wszystkich etapach edukacyjnych, nie musimy więc dobijać się o dodatkowe przedmioty. Taka sytuacja jest tylko w niewielu krajach i zazdroszczą nam tego inni – jest to ewenement na skalę światową. Można oczywiście narzekać, że jest mało godzin. Nie upierajmy się jednak, by dostać ich więcej, bo … możemy stracić to, co szkoły już mają. Wykorzystajmy dobrze ten czas, który już jest na wydzielone zajęcia informatyczne, ale również na innych przedmiotach, w tym tak podatnych na technologię, jak technika, matematyka, czy fizyka.
2. Jeśli są wydzielone przedmioty informatyczne, to oczywiście w szkołach **są nauczyciele**, którzy je prowadzą. A zatem, wystarczy, chociaż jest to olbrzymie wyzwanie, przygotować nauczycieli na proponowane zmiany; o tym piszę w innym miejscu w tych materiałach, przedstawiając standardy przygotowania nauczycieli informatyki.

3. W szkołach **jest sprzęt** do wydzielonych zajęć informatycznych i z powodzeniem może być wykorzystany do realizacji nowej podstawy.
4. **Środowiska programistyczne** pojawiają się jak grzyby po deszczu i w większości **są otwarte i darmowe**. Co więcej, jak Scratch, wręcz zapraszają uczniów do łączenia się społeczności uczących się na całym świecie – propagując *computational participation*.
5. I najważniejsze motywacja dla proponowanych zmian – są **rzesze uczniów**, zaangażowanych w rozwój swoich umiejętności programowania, którym szkoła powinna wyjść naprzeciw. Ci uczniowie uczestniczą dzisiaj w wielu zajęciach pozaszkolnych – warsztatach z programowania i z robotyki – jak i zajęciach, które w szkole są uzupełnieniem regularnych zajęć. Wśród tych ostatnich można wymienić konkursy informatyczne o zasięgu krajowym (Bóbr, olimpiady informatyczne, konkursy programistyczne), jak i niemal w każdy województwie. Wiele inicjatyw jest realizowanych poza szkołami, jak na przykład zajęcia w ramach Godziny Kodowania (code.org; http://godzinakodowania.pl).

3. Propozycja zmian

Polecam lekturę całego tekstu proponowanej podstawy przedmiotów informatycznych, jest dostępny w wielu miejscach. Tutaj zwracam jedynie uwagę na przyjęty opis **Wymagań ogólnych** – są one **identyczne dla wszystkich etapów** kształcenia, czyli faktycznie, są ponad podziałami edukacji na etapy kształcenia. Wymagania szczegółowe zaś dla poszczególnych etapów kształcenia są sformułowane z uwzględnieniem wieku uczniów. ich przygotowania, jak i ewentualnych zainteresowań. Oto te cele ogólne[1]:

> Cele kształcenia informatycznego mają ogólną postać i odnoszą się do wszystkich etapów edukacyjnych i typów szkół. Szczegółowa ich interpretacja jest zapisana w Treściach nauczania – wymaganiach szczegółowych dla poszczególnych etapów kształcenia i typów szkół.
>
> I. **Rozumienie, analizowanie i rozwiazywanie problemów** na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.
>
> II. **Programowanie i rozwiazywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych:** układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

[1] Propozycja wspólnych Wymagań (celów) ogólnych dla całej podstawy programowej (*curriculum*), jak i realizacja procesu rozwiązywania problemów, uwzględniającego programowanie, zaproponowana w p. 5.1, zostały docenione przez zespół roboczy, działający przy TC 3 on Education, IFIP, patrz [9].

III. **Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi**, w tym: znajomość zasad działania urządzeń cyfrowych i sieci komputerowych oraz wykonywania obliczeń i programów.
IV. **Rozwijanie kompetencji społecznych**, takich jak: komunikacja i współpraca w grupie w tym w środowiskach wirtualnych, udział w projektach zespołowych oraz organizacja i zarządzanie projektami.
V. **Przestrzeganie prawa i zasad bezpieczeństwa.** Respektowanie prywatności informacji i ochrony danych, praw własności intelektualnej, etykiety w komunikacji i norm współżycia społecznego; ocena zagrożeń związanych z technologią i ich uwzględnienie dla bezpieczeństwa swojego i innych.

Trzy uwagi są tutaj na miejscu:

- Za zaproponowaną konstrukcją podstawy programowej stoi, wręcz narzucająca się **spiralna metoda** rozwoju wiedzy, pojęć i umiejętności uczniów, uwzględniająca poziom kształcenia, wiek i potrzeby uczniów, jak i rozwój technologii.
- Taki opis ogólnych celów kształcenia jest „nieczuły" na zmiany w systemie edukacji – wymagania szczegółowe w podstawie mogą być „przykrojone" do dowolnego podziału lat w szkole na etapy edukacyjne.
- Zauważmy wreszcie kolejność: „rozumowanie na bazie logicznego i abstrakcyjnego myślenia" (Cel I) występuje przed „programowaniem i rozwiazywaniem problemów z wykorzystaniem komputera" (Cel II).

Te kwestie komentujemy bardziej szczegółowo w następnym rozdziale oraz ilustrujemy w ostatni rozdziale.

4. Wyzwania

Programowanie i informatyka przez 12 lat w szkole?

Dzisiejszy entuzjazm najmłodszych, udzielający się im przy programowaniu, jeszcze nie gwarantuje, że będzie to ich interesować i angażować przez wszystkie 12 lat w szkole. Musimy myśleć o tym już teraz, planując pilotaż na wybranych etapach i w wybranych klasach. Cele ogólne – takie same – przypisane do kolejnych lat pobytu w szkole, każą myśleć o wszystkich etapach jednocześnie, a na poszczególnych etapach – uwzględniać wcześniejsze i późniejsze etapy. To jedno z wyzwań stojących przed nauczycielami – odnosimy się do tego w innym tekście w tych materiałach.

Dla sukcesu proponowanych zmian na przestrzeni 12 lat decydujące mogą być dwa pierwsze etapy szkoły podstawowej, dzisiejsze zajęcia komputerowe – pod obecna podstawą, nauczyciele na ogół rzadko sięgają po programowanie i główna uwaga, ich i uczniów, jest skupiona na technologii – na korzystaniu z gotowych aplikacji. Taką sama diagnozę stawia wielu innych autorów w tych materiałach, np.

P. Perekietka i zespół Ryszarda Szubartowskiego. Ci ostatni wręcz twierdzą, że w gimnazjum może być już za późno, by wzniecić u uczniów zainteresowanie programowaniem.

Informatyka kluczem do dobrobytu?

Jednym z „napędów" zmian w kształceniu informatycznym na świecie jest oceny potrzeb na stanowiskach informatycznych, w tym programistów – w 2020 roku, obecny system kształcenia (w szkołach i w uczelniach) w USA i w Europie nie będzie w stanie zaspokoić potrzeb, szacowanych na dwa razy więcej, niż uczelnie są w stanie „wyprodukować" informatyków. Skąd ich wziąć? Takie same są potrzeby również w Azji, podróże informatyków nie będą więc w stanie wiele zmienić. Trzeba więc zacząć od podstaw, czyli od edukacji, i to na najniższym szczeblu.

Kształcenie informatyczne przez 12 lat może jeszcze nie zapewnić w tym sukcesu. Należy nie „przegapić" momentu w szkole, w którym możemy głębiej zainteresować uczniów informatyką i programowaniem tak, by obrali i podążali ścieżką kształcenia informatycznego na dalszych etapach, kierując się ku karierze zawodowej informatyka i programisty. Wydaje się, że takim momentem w edukacji szkolnej są lata w gimnazjum, a więc po 5-7 latach szkolnej edukacji, w wieku 12-15 lat.

5. Wybrane odpowiedzi na niektóre wyzwania

5.1. Informatyka, programowanie, rozwiązywanie problemów

Kolejność i zawartość Wymagań ogólnych w podstawie (patrz rozdz. 3) jest odpowiedzią na pytanie o miejsce (umiejętności) programowania w kształceniu – programowanie występuje jako elementem szeroko rozumianego kształcenia informatycznego, czyli kształtowania umiejętności stosowania informatycznych metod, w tym programowania komputerów, w procesie rozwiązywania problemów z różnych dziedzin. Programowanie to w pewnym sensie język informatyki – program komputerowy jest medium komunikacyjnym między człowiekiem a komputerem. Oczywiście, przy jego tworzeniu nabywa się i rozwija, jak i stosuje wiele kompetencji, jednak nacisk powinien być kładziony na cel kształtowania umiejętności programowania, którym jest rozwiazywanie problemów z pomocą komputera. Wielu autorów w tych materiałach podkreśla, że informatyka to nie programowanie (P. Perekietka) i jak nie zmienić informatyki w programowanie (A. Jeleńska).

Mając na uwadze kształcenie informatyczne przez cały okres pobytu uczniów w szkole, a nawet dłużej, warto uzmysłowić sobie, że skupienie uwagi na programowaniu, a nie na rozwiązywanych problemach, może doprowadzić do podobnego znużenia uczniów, jakie obserwujemy przy posługiwania się (bez celu) biurowymi

aplikacjami komputerowymi. Przykład: Ćwiczenie z e-podręcznika do informatyki: Napisz jakieś zdanie i skopiuj je 5 razy. Przypomina mi to polecenie nauczyciela z moich lat szkolnych: Usiądź w ostatniej (oślej) ławce i napisz 100 razy „Nie będę przeszkadzał na lekcji". Zniechęcenie do programowania przerodzi się w zniechęcenie do informatyki.

W tym kontekście warto jeszcze podpatrzyć, jak robią to inni, w innych krajach. Hasłem popularnego serwisu code.org (u nas to http://godzinakodowania.pl) jest: **Informatyka to coś więcej niż kodowanie** (tutaj w znaczeniu programowanie). Z kolei, przedmiot, którym objęci są wszyscy uczniowie w Wielkiej Brytanii, to *computing* (ma szersze znaczenie niż *computer science*, informatyka), a w Stanach Zjednoczonych Prezydent Barack Obama ogłosił narodowy program „*Computer Science for ALL*". Te inicjatywy w obu tych krajach obejmują również naukę programowania.

Nasze doświadczenia i doświadczenia bardzo wielu nauczycieli informatyki podpowiadają następującą kolejność etapów na zajęciach z informatyki, uwzględniającą powyższą dyskusję:

problem ⇒ **pojęcia** ⇒ **algorytm** ⇒ **program**

Znaczenie tych etapów:

- uczniowie stają przed **sytuacją problemową** (zamierzoną przez nauczyciela, służącą do realizacji przyjętych celów zajęć); mogą to być gry, łamigłówki, posłużenie się konkretnymi obiektami, które najlepiej, jeśli mają jakieś znaczenie dla uczniów, budowanie robotów i sterowanie nimi, problemy z różnych przedmiotów;
- uczniowie stosują **podejście heurystyczne**[2] do znalezienia rozwiązania, posługują się przy tym **abstrakcją**[3] (każdy program komputerowy jest elementem pewnej abstrakcji), w trakcie pojawiają się **pojęcia** (zamierzone przez nauczyciela);
- uczniowie dochodzą do sposobu rozwiązania problemu – **algorytmu**;
- i wreszcie, znalezione rozwiązanie uczniowie **programują**.

---

[2] Ojcem heurystyki jest George Polya, autor popularnej książeczki, *How to solve it*? [2]. Rozumowanie heurystyczne nie jest traktowane jako ostateczne i ścisłe, ale jako prowizoryczne, jego celem jest odkrycie rozwiązania danego problemu. Buduje się je na doświadczeniach w rozwiazywaniu zadań i obserwowaniu innych ludzi rozwiązujących zadania

[3] Według J. Wing (wprowadziła pojęcie myślenia komputacyjnego, 2006), myśleć jak informatyk, znaczy coś więcej niż umieć programować – wymaga to posługiwania się abstrakcją na wielu poziomach. Z perspektywy konstruktywizmu, poznawanie nowych pojęć to konstruowanie umysłowych obiektów (struktur) abstrakcyjnych i później manipulowanie nimi w umyśle

Komputer, gotowe aplikacje, zasoby sieciowe itp. – mogą pojawić się na dowolnym etapie, w dowolnym momencie, jako wynik decyzji uczniów lub nauczyciela. Powyższe etapy mogą się przeplatać i trwać w zależności od sytuacji problemowej i podjętych przez uczniów działań.

W procesie rozwiązywania problemów należy dopuszczać wszelkie formy aktywności, w uzupełnieniu tekstów pisanych (patrz [7]):

- **wizualne** uczenie się (to posługiwanie się obiektami graficznymi, abstrakcyjnymi i fizycznymi modelami, obrazkowe programowanie, roboty);
- **słuchowe** uczenie się (to rozmowy, dyskusje, w grupach i całą klasą;
- **kinestetyczne** uczenie się (to fizyczne aktywności uczniów)

Dopuszczamy też, by do głosu dochodziły inteligencje wielorakie (H. Gardner) – różne wrażliwości, zdolności i umiejętności: logiczno-matematyczne, językowe, przyrodnicze, muzyczne, przestrzenne, cielesno-kinestetyczne, emocjonalne.

5.2. Programowanie kolejnym narzędziem

Na programowanie można spojrzeć, jak na umiejętności korzystania z innych gotowych aplikacji, np. z pakietu Office. Oczywiście, programowanie w większym stopniu wymaga od użytkownika umiejętności abstrakcyjnego i logicznego myślenia, kreatywności, ale przecież praca nad tekstem, tworzenie ilustracji i multimediów, wykonywanie rachunków i zapisywanie algorytmów w arkuszu, tworzenie baz danych i korzystanie z nich, te wszystkie działania są również z pogranicza logicznego myślenia i kreatywności (w materiałach jest praca Panów Sławomira Hermy i Bartłomieja Żywczaka o programowaniu w arkuszu kalkulacyjnym). Inna sprawą jest, na ile nasi uczniowie „uruchamiają" te procesy w trakcie korzystania z tych aplikacji, na ile nauczyciele stawiają przed nimi zadania/problemy, wymagające takich aktywności.

Programowaniem interesuje się wielu nauczycieli różnych przedmiotów, uczestniczą w szkoleniach, programują wraz ze swoimi uczniami. Tematy tych programów, zwykle tworzonych w środowisku wizualno-blokowym, pochodzą z obszarów pierwszych zainteresowań nauczycieli: matematyki, historii, języka polskiego i obcego, fizyki, plastyki – wystarczy obejrzeć galerię projektów w języku Scratch.

Można pokusić się o postawienie tezy, że obecnie wszyscy nauczyciele otrzymują kolejne, po aplikacjach biurowych i sieciowych, środowisko komputerowe, które mogą wykorzystywać w swoich dziedzinach (przedmiotach) kształcenia. Uważa się, że środowisko programowania w języku Scratch oferuje dzisiaj możliwości tworzenia prezentacji znacznie przewyższające ofertę programu PowerPoint – prezentacje mogą być interaktywne, dynamiczne, multimedialne. Odsyłam do pracy w tych materiałach Pań Ewy Kędrackiej-Feldma i Małgorzaty Rostkowskiej, które zgłębiają szanse dla nowej podstawy kształcenia informatycznego na zamianę

dydaktyki innych przedmiotów, przywołując do tego odpowiednią metodologię postępowania, będącą odmianą podobnego podejścia z połowy lat 90' XX wieku, które stosowaliśmy na przykład przy organizacji studiów podyplomowych i szkoleń dla nauczycieli.

Informatyka czy technologia nigdy nie obwarowywały się, a wręcz przeciwnie, otwierały swoje arsenały dla innych dziedzin, dla wszystkich przedmiotów szkolnych. Podobnie jest teraz ze środowiskami programowania. Programowanie jest jednak umiejętnością znacznie bardziej złożoną i od pewnego etapu wymaga dobrego przygotowania informatycznego, takich umiejętności uczniowie mogą nabyć na lekcjach informatyki, a nauczyciele innych przedmiotów mogą z tego tylko korzystać.

5.3. Informatyka bez komputera?

Naszkicowany tok zajęć, poświęconych rozwiązywaniu zadań z pomocą komputera, wcale nie sugeruje, ze uczeń cały czas ma dysponować i posługiwać się komputerem. Wprost przeciwnie, powinien poza komputerem „przygotować" się do posłużenia się nim. W szczególności, przygotować się do napisania programu, będącego rozwiązaniem postawionego zadania, sytuacji problemowej. To przygotowanie może przybierać różne formy, w zależności od rozważanej sytuacji problemowej. Czas trwania wymienionych wcześniej etapów rozwiązywania problemu na ogół będzie zależał od problemu i od obranej drogi rozwiązywania.

Informatyki bez komputera (ang. *compurer science unplugged*) nie należy traktować dogmatycznie. Na poszczególnych etapach rozwiazywania problemów uczniowie mogą sięgnąć po komputer, jeśli tylko ma to im pomóc w procesie rozwiazywania i znalezieniu rozwiązania, np. posługując się programami edukacyjnymi, których jest wiele – polecam zgromadzone u mnie [5]. Na zajęciach z uczniami w ramach Uniwersytetu Dziecięcego często przeplataliśmy zajęcia bez komputera aktywnościami przy komputerze, osłabiając i proponując tym samym „osłabioną" wersję informatyki bez komputerów, bliższą zajęciom informatycznym. Podobnie, takie mieszane podejście oferuje konkurs Bóbr [1]. Zadania Bobrów i dla Bobrów w każdym wieku, motywujące ich do logicznego myślenia w obszarze komputacyjnego myślenia, są rozwiązywane na komputerze, często są to zadania interaktywne, ale komputer nie pomaga w tym uczniom, a jedynie służy do sprawnego zaprezentowania zadań, zebrania rozwiązań i przeprowadzenia całego konkursu.

Rozwój kompetencji informatycznych bez komputera, w szczególności myślenia komputacyjnego, szczegółowo omawia Pan Paweł Perekietka, który zajmuje się m.in. przekładem scenariuszy takich zajęć na język polski. Jego artykuł dostarcza wielu przykładów rozwoju pojęć i metod informatycznych poza komputerem.

Chciałbym zwrócić uwagę na kontekst historyczny, mocno uwypuklony w pracy Pana Pawła, który, mam nadzieję przekona Państwa, iż myślenie komputacyjne „uprawiało" wielu myślicieli, odkrywców, innowatorów i twórców nawet w odległej historii rozwoju ludzkości, jak na przykład Morse, prekursor dzisiejszej kryptografii.

Sięgnę jednak czasów nam współczesnych. W latach 60-70, gdy do komputerów było daleko, nie nosiliśmy przy sobie nawet suwaków i stroniliśmy od mechanicznych maszyn do rachowania, a kalkulatory dopiero się rodziły, zajęcia z algorytmiki i programowania były głównie „informatyką bez komputera". Dla przykładu, stabilne związki, dla których algorytm jest opisany w [4], tworzyliśmy między sobą, studenci z prowadzącymi, i do dzisiaj go pamiętamy, chociaż nie zawsze stosowaliśmy go w życiu. Nie raz się zdarzyło, że program napisany dla tak „przerobionego" algorytmu, uruchamiał się bez błędu za pierwszym razem.

Dzisiaj, nawet w sytuacji, gdy komputer leży przed nami, przed naszymi uczniami, warto odsunąć go nawet na dłuższą chwilę, by przygotować się do posłużenia się nim. Również dzisiaj takie podejście może być lekarstwem na brak komputerów, by każdy uczeń siedział przed swoim. Przy okazji uczniowie mogą sobie pomagać w dojściu do rozwiązania komputerowego.

6. Porządkowanie przez 12 lat w szkole

W tym rozdziale chciałbym krótko przekonać Państwa, że ten klasyczny problem sortowania – będę raczej porządkował – może być problemem-rzeką, znakomicie nadającym się na zajęcia przez 12 lat w szkole. Inne spojrzenie na zagadnienie sortowania – szalenie ważne w informatyce szkolnej – zaprezentuje Państwu prof. Krzysztof Diks. Ten problem może być wehikułem dla wielu pojęć i metod informatyki, przystępnym dla uczniów na różnych etapach kształcenia.

Poniżej, dla poszczególnych etapów edukacyjnych wymieniam przykładowe sytuacje problemowe, aktywności uczniów, narzędzia (przybory), kształtowane pojęcia informatyczne (czasem bez pełnej świadomości), algorytmy, programy.

K-3 – przedszkole, klasy 1-3

Sytuacja: porozrzucane karty z obrazkami zwierząt, owoców, …
Cel: pogrupujcie według własnego uznania
Narzędzia: karty z obiektami, rzeczywiste obiekty, …

Sytuacja: różne rzeczy, odpady
Cel: posegregujcie według rodzaju – segregowanie odpadów
Narzędzia: własne ręce, robot
Uwaga: polecam warsztaty na ten temat z udziałem robotów Dash & Dot

Pojęcia (informatyka): metoda kubełkowa, haszowanie (na później)

1-3:

Sytuacja: obrazki, np. zwierząt czworonożnych, poukładane w rzędzie
Cel: ustawcie według wagi ciała
Narzędzia: własne ręce, robot
Wsparcie: zabawy kinestetyczne, zadania z konkursu Bóbr (ustawianie ołówków w kolejności)
Metodyka: abstrakcyjne myślenie (waga – abstrakcją), odkrywanie własnych sposobów

Pojęcia (informatyka): porządek (kolejność), przestawianie elementów sąsiednich i na początek

4-6:

Sytuacje: różnorodne sytuacje, związane z porządkowaniem i wyborem
Cel: różnorodny kontekst występowania uporządkowania i sposób porządkowania
Wsparcie: zadania z konkurs Bóbr: znajdowanie najlżejszej monety, wybór ciągu uporządkowanego, sieć sortująca
Programowanie: własny program w środowisku wizualno-blokowym, zabawa i ćwiczenia z gotową aplikacją do porządkowania

Pojęcia (informatyka): różnorodne konteksty porządkowania i metody dostosowane do kontekstu

4-6, Gim:

Sytuacje: organizacja rozgrywek w szkole, indywidualnych lub zespołowych, wybór obiektu naj …
Cel: znajdź najlepszy/największy/najmniejszy/naj… element
Wsparcie: kinestetyczna gra, plansza klasowych/szkolnych rozgrywek
Programowanie: programy w języku Python dla wybranych zagadnień, gotowe programy, jako ilustracja

Pojęcia (informatyka): przeszukiwanie liniowe, turniejowe – liczba porównań

4-6, Gim:

Sytuacje: uporządkowany ciąg obiektów, ale również liczb – ukryte
Cel: znajdź, czy ciąg zawiera wybrany element, możliwie najmniejszą liczba pytań
Wsparcie: zabawa w zgadywanie liczby, zadania z Bobra (bardzo wiele)
Programowanie: Uwaga: napisanie poprawnego programu dla przeszukiwania binarnego nie jest takie proste, można posłużyć się gotowym

Pojęcia (informatyka): znaczenie porządku dla wyszukiwania informacji (danych), łatwo znaleźć, gdy szukamy przez podział ciągu wśród 1000 uporządkowanych elementów, to prób jest tylko 10

Gim:

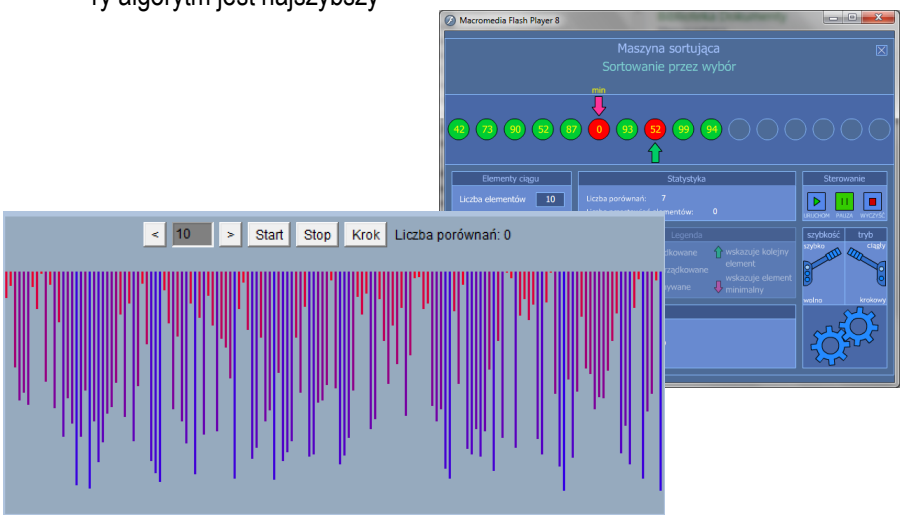
Sytuacja: ciąg liczb
Cel: uporządkuj
Wsparcie: programy demo – patrz poniżej, Godzina Kodowania
Programowanie: własne programy w języku Python dla wybranych algorytmów

Pojęcia (informatyka): iteracja: najmniejszy na początek, przestawić elementy w złej kolejności, pierwsze algorytmy porządkowania,

Gim, LO, LO rozszerzenie:

Sytuacja: ciągi liczb, losowe i specjalne
Cel: uporządkuj możliwie najszybciej

Pojęcia (informatyka): elementy komputerowych implementacji: iteracja, scalanie, rekurencja, porównanie efektywności wybranych metod porządkowania: przez wybór – stała liczba działań, bąbelkowa – szybka na mało nieuporządkowanym ciągu, porównanie czasów obliczeń – dyskusja, który algorytm jest najszybszy



Ilustracje: Aplikacje ze strony http://mmsyslo.pl/Materialy/Oprogramowanie, służące do wizualizacji algorytmów porządkowania i do przeprowadzania z nimi eksperymentów

Literatura

1. Konkurs Bóbr: http://bobr.edu.pl
2. Polya G., *Jak to rozwiązać*, WN PWN, Warszawa 2010 (Oryginał, 1945)
3. Sysło M.M., *Algorytmy*, Helion, Gliwice 2016 (wydawana od 1997 roku przez WSiP, Warszawa)
4. Sysło M.M., *Piramidy szyszki i inne konstrukcje algorytmiczne*, Helion, Gliwice 2015 (pierwsze wydanie – WSiP, Warszawa 1998)
5. Sysło M.M., Oprogramowanie edukacyjne do algorytmiki, http://mmsyslo.pl/Materialy/Oprogramowanie
6. Sysło M.M., *Kierunki rozwoju edukacji wspieranej technologią. Nowe technologie w edukacji. Propozycja strategii i planu działania na lata 2014-2020*, Wrocław, Toruń 2014.
7. Sysło M.M., Kwiatkowska A.B., Playing with Computing at a Children's University, WiPSCE '14, ACM, Berlin 2014
8. Sysło M.M., *Rozwój edukacji informatycznej w Polsce na tle historycznym*, książka w przygotowaniu dla PTI.
9. Webb M. et al., Computer science in the curriculum: issues and challenges, praca zgłoszona na konferencję ISSEP 2016.