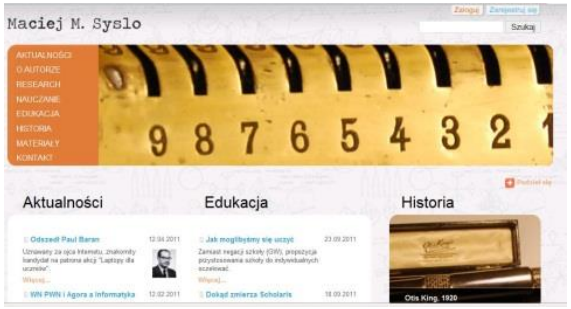
Informatyka i programowanie przez wszystkie lata w szkole

Proponowana podstawa programowa kształcenia informatycznego i jej realizacja

Maciej M. Sysło
UMK Toruń, UWr Wrocław
syslo@mat.uni.torun.pl, …@ii.uni.wroc.pl
http://mmsyslo.pl



Polska … przed USA

Koniec grudnia, 2015:

- szybki Internet do wszystkich szkół (MC)
- nowa podstawa programowa informatyki, a w niej programowanie dla wszystkich uczniów

Koniec stycznia, 2016:

Prezydent Obama ogłasza Computer Science for ALL

i obiecuje znaleźć na to $ 4 000 000 000

W Wielkiej Brytanii: Koniec z ICT w szkołach ?

ICT znika z podstawy programowej i robi miejsce dla informatyki

Minister edukacji w UK

To było w 2012 roku
Od września 2014 do szkół UK weszła informatyka i programowanie dla wszystkich uczniów – Computing

Badania EuroStat

	Copied or moved a file or folder		Used basic arithmetic formulas in a spreadsheet		Created electronic presentations		Written a computer program	
	Aged 16-74	Aged 16-24	Aged 16-74	Aged 16-24	Aged 16-74	Aged 16-24	Aged 16-74	Aged 16-24
EU27	63	89	43	67	31	59	10	20
Belgium	68	92	46	67	35	70	11	20
Bulgaria	41	76	22	47	6	18	2	5
Czech Republic	60	89	43	74	18	42	5	11
Denmark	79	95	67	88	50	88	11	19
Germany	72	94	44	80	33	67	9	18
Estonia	59	91	47	75	25	48	9	21
Ireland	60	82	44	54	21	36	9	(13)
Greece	47	88	34	65	23	55	8	17
Spain	58	84	41	66	33	66	12	27
France	67	85	49	74	38	63	11	17
Italy	54	85	38	61	23	50	9	18
Cyprus	53	92	41	77	29	65	6	12
Latvia	61	97	46	87	32	75	7	18
Lithuania	57	97	42	82	29	68	8	20
Luxembourg	80	96	62	73	50	75	16	(21)
Hungary	63	92	48	81	20	45	11	25
Malta	59	93	44	74	30	63	8	(21)
Netherlands	81	95	54	63	55	89	9	12
Austria	75	99	56	87	43	84	13	30
Poland	52	94	33	70	16	47	6	16
Portugal	57	96	42	78	32	78	7	18
Romania	38	72	20	46	8	18	5	16
Slovenia	61	97	48	85	36	85	6	(16)
Slovakia	70	97	57	87	41	84	6	13
Finland	77	95	61	76	52	84	26	37
Sweden	73	94	58	83	51	72	24	34
United Kingdom	72	94	51	72	38	61	13	25
Iceland	82	94	73	86	55	88	15	20
Norway	68	89	67	85	61	86	18	(20)

Polacy a umiejętność programowania

średnia europejska: 20

Polscy uczniowie i studenci: 16

22 miejsce w Europie

PISA 2012

	Performance in problem solving				Relative performance in problem solving,	Performance in problem solving, by process		Performance in problem solving, by nature of the problem situation	
	Mean score in PISA 2012	Share of low achievers (below Level 2)	Share of top performers (Level 5 or 6)	Gender difference (boys - girls)	compared with students around the world with similar performance in mathematics, reading and science	Solution rate on tasks measuring **acquisition of knowledge**	Solution rate on tasks measuring **utilisation of knowledge**	Solution rate on items referring to a **static** problem situation	Solution rate on items referring to an **interactive** problem situation
	Mean score	%	%	Score dif.	Score dif.	Percent correct	Percent correct	Percent correct	Percent correct
OECD average	500	21.4	11.4	**7**	-7	45.5	46.4	47.1	43.8
Singapore	562	8.0	29.3	**9**	2	62.0	55.4	59.8	57.5
Korea	561	6.9	27.6	**13**	14	62.8	54.5	58.9	57.7
Japan	552	7.1	22.3	**19**	11	59.1	56.3	58.7	55.9
Macao-China	540	7.5	16.6	**10**	8	58.3	51.3	57.0	51.7
Hong Kong-China	540	10.4	19.3	**13**	-16	57.7	51.1	56.1	52.2
Shanghai-China	536	10.6	18.3	**25**	-51	56.9	49.8	56.7	50.3
Chinese Taipei	534	11.6	18.3	12	-9	56.9	50.1	56.3	50.1
Canada	526	14.7	17.5	**5**	0	52.6	52.1	52.7	50.5
Australia	523	15.5	16.7	2	7	52.3	51.5	52.8	49.9
Finland	523	14.3	15.0	**-6**	-8	50.2	51.0	52.1	47.7
England (United Kingdom)	517	16.4	14.3	6	8	49.6	49.1	49.5	47.9
Estonia	515	15.1	11.8	5	-15	46.8	49.5	49.7	45.6
France	511	16.5	12.0	5	5	49.6	49.4	50.3	47.6
Netherlands	511	18.5	13.6	5	-16	48.2	49.7	50.4	46.5
Italy	510	16.4	10.8	**18**	10	49.5	48.0	49.5	46.8
Czech Republic	509	18.4	11.9	8	1	45.0	46.9	46.2	44.4
Germany	509	19.2	12.8	**7**	-12	47.5	49.5	49.4	46.3
United States	508	18.2	11.6	3	10	46.5	47.1	46.6	45.9
Belgium	508	20.8	14.4	**8**	-10	47.0	47.5	48.3	45.4
Austria	506	18.4	10.9	**12**	-5	45.7	47.4	48.3	43.0
Norway	503	21.3	13.1	-3	1	47.7	48.1	49.4	44.5
Ireland	498	20.3	9.4	5	-18	44.6	45.5	44.4	44.6
Denmark	497	20.4	8.7	**10**	-11	44.2	48.1	47.9	42.3
Portugal	494	20.6	7.4	**16**	-3	41.6	45.7	44.0	42.0
Sweden	491	23.5	8.8	-4	-1	45.2	44.6	47.7	41.6
Russian Federation	489	22.1	7.3	**8**	-4	40.4	43.8	43.8	39.7
Slovak Republic	483	26.1	7.8	**22**	-5	40.5	43.2	44.2	38.8
Poland	481	25.7	6.9	0	-44	41.3	43.7	44.1	39.7
Spain	477	28.5	7.8	2	-20	40.0	42.3	42.3	39.8
Slovenia	476	28.5	6.6	-4	-34	37.8	42.3	42.9	36.7
Serbia	473	28.5	4.7	**15**	11	37.7	40.7	40.3	36.8
Croatia	466	32.3	4.7	**15**	-22	35.2	40.5	39.3	35.6
Hungary	459	35.0	5.6	3	-34	35.2	37.6	38.2	33.9
Turkey	454	35.8	2.2	**15**	-14	32.8	36.0	35.8	32.7
Israel	454	38.9	8.8	6	-28	38.7	37.0	39.7	35.6
Chile	448	38.3	2.1	**13**	1	30.9	35.2	34.9	31.8
Cyprus*	445	40.4	3.6	**-9**	-12	33.6	34.8	37.0	31.4
Brazil	428	47.3	1.8	**22**	7	28.0	32.0	29.8	29.1
Malaysia	422	50.5	0.9	**8**	-14	29.1	29.3	30.1	27.4
United Arab Emirates	411	54.8	2.5	**-26**	-43	28.4	29.0	29.9	27.1
Montenegro	407	56.8	0.8	**-6**	-24	25.6	30.0	30.3	25.1
Uruguay	403	57.9	1.2	**11**	-27	24.8	27.9	27.5	24.8
Bulgaria	402	56.7	1.6	**-17**	-54	23.7	26.7	28.4	22.3
Colombia	399	61.5	1.2	**31**	-7	21.8	27.7	26.3	23.7

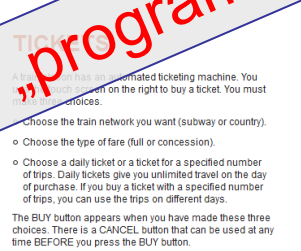
Polska: Poniżej średniej europejskiej we wszystkich kategoriach

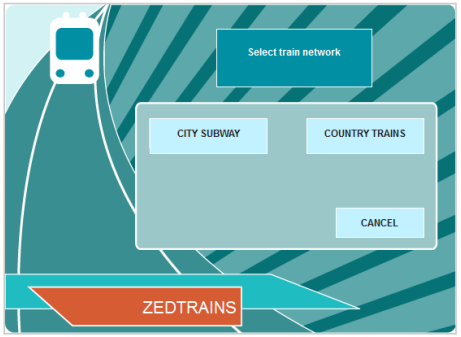
Jedna z motywacji

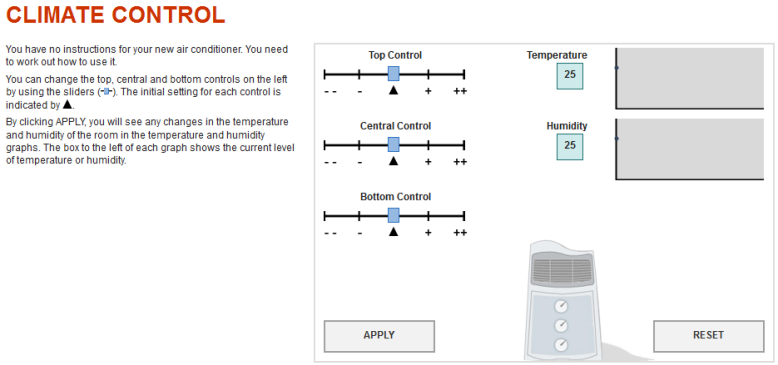
Dlaczego nasi uczniowie wypadli źle (29 miejsce na 32 kraje) w badaniach PISA w zakresie rozwiązywania problemów, programowania urządzeń cyfrowych.

Testy: najkrótsze drogi klimatyzator biletomat

Rozwiązywanie problemów i „programowanie”







Inna motywacja – potrzeby i kompetencje

- 2020 w USA: potrzebnych będzie ponad 1 mln osób z wykształceniem informatycznym, a uczelnie opuści tylko 400 tys.
- Podobnie w UK, UE i w Polsce
- ale ok. 30 tys. absolwentów starało się na kierunki informatyczne, na ogół nie przygotowanych do studiowania informatyki – tylko 4 tys. zdawało maturę z informatyki
- stąd duży odsiew na I roku studiów (ponad 50%) – zły, nie przygotowany wybór do studiowania

Generalnie:

- Kompetencje informatyczne = kompetencje rozwiązywania problemów z pomocą komputera = kompetencje rozwiązywania jakichkolwiek problemów niemal w każdej dziedzinie

Zapowiedź …

Z końca grudnia 2015:

- szybki Internet do wszystkich szkół (MC)
- nowa podstawa programowa informatyki, a w niej programowanie dla wszystkich uczniów:
  - 2016/2017 – pilotaż
  - 2017/2018 – na dobre we wszystkich szkołach

MUSIMY BYĆ GOTOWI! W pewnym sensie jesteśmy:

- na każdym etapie edukacyjnym istnieją przedmioty informatyczne
- w szkołach pracują nauczyciele tych przedmiotów
- szkoły są wyposażone w podstawowy sprzęt informatyczny
- środowiska programistyczne są powszechnie dostępne i bezpłatne
- duże zaangażowanie uczniów i gotowość do udziału w zajęciach informatycznych/programistycznych

Zapowiedź …

Propozycja godzin (10.10.2016):

	Klasa	Liczba godzin
	1	1
	2	1
	3	1
	4-8	po 1
LO	1	1
	2	1
	3	1
	4	0

Rozszerzenie od 1-szej klasy LO:

1-4	po 2

Propozycja Rady przy MEN

Nowa podstawa przedmiotu informatyka dla każdego poziomu edukacyjnego, adresowana do wszystkich uczniów. Założenia:

- informatyka jest czymś więcej niż tylko programowaniem
- myślenie komputacyjne, a w jego ramach programowanie, to czwarta podstawowa alfabetyzacja obok: czytania, pisania i rachowania (popularne 3R).
- uczniowie poznają podstawy informatyki, nabywają przy tym umiejętność kreatywnego wykorzystania technologii w realizacji swoich pomysłów i rozwiązywaniu problemów
- poznanie i korzystanie z informatyki – myślenia komputacyjnego – jest fundamentem dla poznania świata, jak i przyszłego dobrobytu i pełnego uczestnictwa w życiu osobistym, zawodowym i społecznym

Nie ma innej dziedziny, która spinałaby tak wiele innych dziedzin.

Propozycja podstawy programowej

Wspólne Cele kształcenia – Wymagania ogólne – dla wszystkich etapów

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.

Technologia

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, netykiety, norm współżycia społecznego, praw własności intelektualnej; ocena i uwzględnienie zagrożeń, związanych z technologią.

Spiralna realizacja na kolejnych etapach

Wyzwania w realizacji kształcenia informatycznego

1. Programowanie od najmłodszych lat – ale co dalej?
   - programowanie musi być w szerszym kontekście: zabaw, gier, zintegrowanych działań
2. Jak podtrzymywać zainteresowanie uczniów programowaniem i informatyką przez 12 lat w szkole?
   - pierwsze lata (1-3 i 4-6) są najważniejsze dla sukcesu na dalszych etapach
3. Rola programowania – elementem rozwiązywania problemów
4. Nie przegapić w szkole momentu głębszego zainteresowania informatyką i programowaniem – dalszą ścieżką rozwoju uczniów w kierunku informatycznym

O czym dalej …

- Miejsce programowania
- Tok zajęć. Podejście: heurystyka i cel: kształtowanie abstrakcji
- Aktywności uczniów
- Inteligencje i myślenie wielorakie
- Przykłady – rozwijające się na kolejnych poziomach edukacji
  - Różne poziomy programowania robota
  - Pomysły Bobrów
  - Godzina kodowania trwająca cały rok
  - Porządkowanie i algorytm Euklidesa przez 12 lat

UWAGI:

1. Jest to krótki kurs dla nauczycieli
2. Głównie propaguję myślenie

Dylemat jajka i kury … ... a może kogut?

Dylemat – miejsce programowania:

Chyba nie:

Kodować by się uczyć (*Code to learn*)

Ale raczej:

Uczyć się programować, jeśli potrzeba (*Learn to code*)

By w przyszłości, ewentualnie:

Programować, by zarabiać (*Code to earn*)

Stare (J. Szczepkowicz):

NN nie miał nic do powiedzenia

NN nauczył się po hiszpańsku

teraz NN nie ma nic do powiedzenia po hiszpańsku

hiszpański = jakikolwiek język programowania

Język programowania, to język komunikacji z komputerem. Trzeba mieć coś do powiedzenia w tej rozmowie z komputerem.

Tok zajęć: problem, pojęcia, algorytm, program

Stąd propozycja toku zajęć z elementami programowania:

- sytuacja problemowa (zamierzona przez nauczyciela) do rozwiązania przez uczniów: gry/aktywności kooperacyjne, łamigłówki z użyciem obiektów, które mają konkretne/realne znaczenie dla uczniów, roboty, problemy z różnych przedmiotów

  podejście do rozwiązania: heurystyka:

  efekt: abstrakcja, pojęcia
- pojawia się: sposób rozwiązania, algorytm
- algorytm, przepis, rozwiązanie można zaprogramować

Komputer, gotowe aplikacje, zasoby sieciowe itp. – mogą pojawić się na dowolnym etapie, w dowolnym momencie – wynikiem decyzji nauczyciela lub uczniów

Dwa ważne pojęcia: abstrakcja i heurystyka

Abstrakcja

- J. Wing (CT), Myśleć jak informatyk, znaczy coś więcej niż umieć programować – wymaga to posługiwania się abstrakcją na wielu poziomach. Program to twór abstrakcyjny.
- Z perspektywy konstruktywizmu, poznanie nowych pojęć polega na skonstruowaniu umysłowych obiektów (struktur), abstrakcyjnych, i później manipulowaniu nimi w umyśle

Heurystyka (George Polya, *Jak to rozwiązać*, 1945)

- Rozumowanie heurystyczne nie jest traktowane jako ostateczne i ścisłe, ale jako prowizoryczne i tylko prawdopodobne, którego celem jest odkrycie rozwiązania danego zadania
- Heurystykę buduje się na doświadczeniu w rozwiazywaniu zadań i obserwowaniu innych ludzi rozwiązujących zadania
- Odpowiednie zadanie, uczeń musi chcieć je rozwiązać.

Aktywności, inteligencje, myślenie – uczniów

Zalecane trzy formy aktywności, w uzupełnieniu tekstów:

- wizualne uczenie się (obiekty graficzne, modele abstrakcyjne i fizyczne, obrazkowe programowanie, roboty)
- słuchowe uczenie się (rozmowy, dyskusje, grupy i cała klasa, …)
- kinestetyczne uczenie się (fizyczne aktywności uczniów)

Inteligencje wielorakie (H. Gardner) – wrażliwości, zdolności, umiejętności:

logiczno-matematyczna, językowa, przyrodnicza, muzyczna, przestrzenna, cielesno-kinestetyczna, emocjonalna (interpersonalna, intrapersonalna).

Myślenie komputacyjne (*mental tools*) – metody umysłowe, rozumowania, związane z rozwiązywaniem problemów, gdy mamy możliwość i przewidujemy posłużenie się komputerem.

Te metody na ogół wywodzące się z informatyki.

SP 1-3, cele ogólne I i II

I. Rozumienie, analizowanie i rozwiazywanie problemów. Uczeń:

1) Porządkuje w postaci sekwencji (liniowo) następujące informacje:
   - obrazki i teksty składające się na historyjki (*storytelling*),
   - polecenia (instrukcje) składające się codzienne czynności,
2) planuje w ten sposób późniejsze ich zakodowanie za pomocą komputera.
3) Tworzy polecenia (sekwencję poleceń) dla określonego planu działania lub dla osiągnięcia celu. W szczególności wykonuje te polecenia w aplikacji komputerowej.

II. Programowanie i rozwiazywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych. Uczeń:

1) Korzysta z przystosowanych do swoich możliwości i potrzeb aplikacji komputerowych, związanych z kształtowaniem podstawowych umiejętności: pisania, czytania, rachowania i prezentowania swoich pomysłów.

TI:

2) Programuje wizualnie proste sytuacje/historyjki według pomysłów własnych i pomysłów opracowanych wspólnie z innymi uczniami.
3) Steruje robotem lub inną istotą na ekranie komputera lub poza komputerem.

Informatyka bez komputera – Wykształcona małpa

1916

Dla 5 x 5

Wykorzystywana na zajęciach Uniwersytetów Dziecięcych

Służy do:

- mnożenia dwóch liczb
- dzielenia dwóch liczb
- rozkładu liczby na czynniki

Z podkładką, może służyć do dodawania

Pojęcia:

- podstawowe operacje matematyczne,
- posługiwanie urządzeniami do liczenia – elektroniczny kalkulator później, tutaj widać, jak są wykonywanie działania
- algorytm

Dzieci były zainteresowane, gdzie można kupić takie urządzenie !!!!

Roboty, które nas słuchają

Poziom 1-3, 4-6 – demonstracja Dash & Dot:

- zabawy ruchowe: odkrywanie, jakie ma możliwości poruszania się, błyskania, wydawania dźwięków) – 1-3
- programowanie „dotykowe” – pod dyktando – 1-3
- programowanie „dotykowe” – własne schematy – 1-3
- programowanie w Blockly – 4-6
- wykorzystanie akcesoriów – np. cymbałki

Wiele innych konstrukcji (warsztaty):

- łamigłówki poza tabletem, wczytywane
- Lego WeDo 2.0, inne Lego, …
- Arduino
- Raspberry Pi

majsterkowanie, mechatronika – połączenie mechaniki z elektroniką

Świetne zajęcia z techniki !

Roboty, które nas słuchają

Pierwsze kroki – sortowanie odpadów

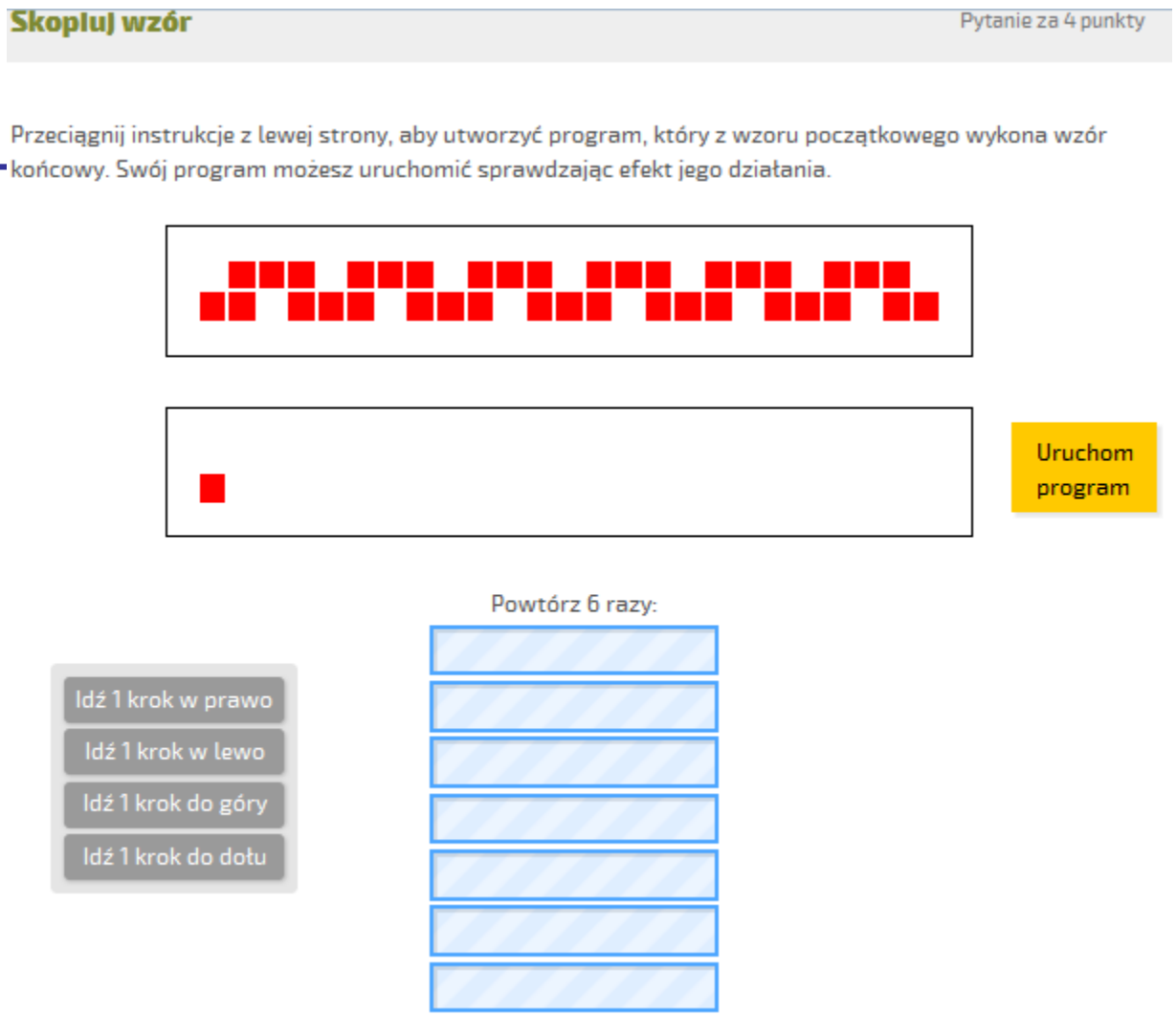
Schemat blokowy do kierowania robotem

Program w Blockly

Propozycje Bobrów

Bóbr 2015, zadanie dla Skrzatów (1-3):

- utwórz program – przeciągnij i upuść
- ale zanim, zauważ powtarzające się motywy (iteracja, redukcja)
- sprawdź program – testowanie
- ewentualnie, popraw program – debugowanie



Godzina kodowania

Godzina kodowania … przez cały rok:

- od przedszkola po uniwersytet – na każdy poziom
- XII.2014 – XII.2015 – 500 tys. uczniów z Polski (10-16 miejsce w świecie) wśród 140 mln.
- w wielu szkołach, na tym bazuje wprowadzenie do programowania w dowolnym języku:
  - bohaterowie uczniów – z gier i realu, gotowe kursy, przekrój wszystkich konstrukcji algorytmicznych w postaci łamigłówek

Program, czy czarna skrzynka?

R.W. Hamming (1959):

The purpose of computing is insight, not numbers

Celem obliczeń nie są liczby, a zrozumienie

Przykłady „czarnych skrzynek” w języku Python:

1. Sortowanie – jak to działa, jaki algorytm jest w sorted?

```
>>> a = [3, 6, 8, 2, 78, 1, 23, 45, 9]    #definiujemy ciąg liczb
>>> sorted(a)            #sortujemy, ale jak?
[1, 2, 3, 6, 8, 9, 23, 45, 78]    #wynik
```

2. Algorytm Euklidesa
   – dlaczego to działa i jak długo?
   np. dla liczb 10^{300} w szyfrowaniu RSA?

```
def EuklidRek(m,n):
  if m > n:
    return EuklidRek(n,m)
  else:
    if m == 0:
      return n
    else:
      return EuklidRek(n % m,m)
```

Porządkowanie … przez 12 lat w szkole, 1

1. K-3:

- Sytuacja: porozrzucane karty z obrazkami zwierząt, owoców itp.

  Cel: pogrupujcie według własnego uznania

- Sytuacja: różne rzeczy, odpady

  Cel: segregowanie według rodzaju

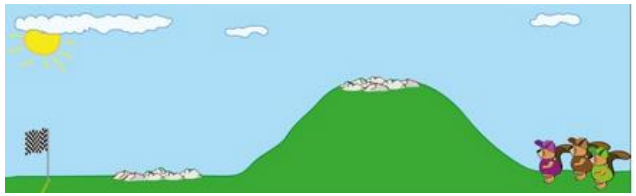
- Informatyka, pojęcia: haszowanie, metoda kubełkowa

2. 1-3:

- Sytuacja: np. ciąg obrazków zwierząt czworonożnych

  Cel: ustawcie według wagi ciała

- Informatyka, pojęcia:– porządek, przestawianie, przestawianie sąsiednich, od najlżejszych
- Metodyka: abstrakcyjne myślenie, odkrywanie własnych sposobów
- Wsparcie, zabawy – Bóbr:



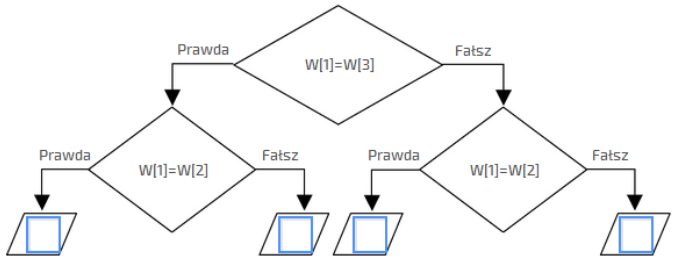
Porządkowanie … przez 12 lat w szkole, 2

3. 4-6:

- Sytuacje: różnorodne
- Cel: różnorodny kontekst występowania uporządkowania i sposób porządkowania
- Informatyka: różnorodne konteksty porządkowania i metody dostosowane do kontekstu

Fałszywa moneta Pytanie za 3 punkty

Mamy 4 monety, 3 prawdziwe i jedną fałszywą; fałszywa ma inną wagę niż prawdziwe. Monetę fałszywą można znaleźć za pomocą dwóch porównań wagi monet na wadze szalkowej. Poniższy schemat blokowy jest zapisem takiego ważenia - bloki w postaci diamentów odpowiadają porównaniu wagi dwóch monet, np. w górnym diamencie porównywana jest waga monet 1 i 3.



Uporządkowania Pytanie za 3 punkty

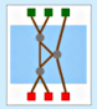
Ciąg liczb jest posortowany **rosnąco**, jeśli dla wszystkich liczb w ciągu każda kolejna liczba jest większa
lub równa od liczby bezpośrednio poprzedzającej ją. Podobnie ciąg jest posortowany **malejąco**, jeśli dla wszystkich liczb w ciągu każda kolejna liczba jest mniejsza lub równa od liczby bezpośrednio poprzedzającej ją. Ciąg nie jest posortowany, jeśli nie jest posortowany ani rosnąco ani malejąco.

Który z podanych ciągów nie jest posortowany?

- `3.01; 3.1; 3.001; 3.101`
- `10.74; 4; -3.5; -10.74`
- `-5; -2.5; -2.05; 2.5`
- `-8.1; -8.01; 8.001; 8.01`

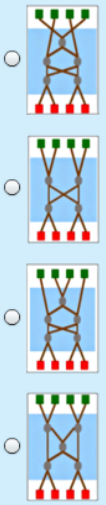
Sieć sortująca

Trzy bobry bawią się nad rzeką w „grę sortującą". Używając kamieni i drewnianych belek budują następującą sieć złożoną z miejsc (czerwonych miejsc początkowych, zielonych miejsc końcowych na brzegach rzeki i szarych miejsc na rzece) oraz połączeń między miejscami (z drewnianych belek).



Bobry używają takiej sieci w następujący sposób: Z jednego miejsca przechodzą po belce ku górze do następnego miejsca. Na miejscu (kamieniu), bóbr czeka na drugiego bobra i mniejszy bóbr wybiera lewe połączenie, a większy bóbr wybiera prawe połączenie. Ciekawe jest, że bez względu na to, jakie pozycje początkowe zajmują bobry (czerwone miejsca), zawsze kończą one na pozycjach końcowych od najmniejszego z lewej strony, do największego z prawej strony.

Nagle, czwarty bóbr postanowił zabawić się z trzema bobrami. Teraz potrzebują one nowej sieci, która będzie porządkować cztery bobry. Próbują one wykorzystać następujące cztery sieci, ale tylko jedna z nich poprawnie je uporządkuje. Która?



Porządkowanie … przez 12 lat w szkole, 3

4. 4-6, Gim

- Sytuacje: organizacja rozgrywek, wybór obiektu naj …

  Cel: znajdź najlepszy/największy/najmniejszy/naj… element
- Informatyka: przeszukiwanie liniowe, turniejowe – liczba porównań, pierwsze programy (Pyhon)
- Wsparcie: kinestetyczna gra, plansza klasowych/szkolnych rozgrywek

5. 4-6, Gim

- Sytuacje: uporządkowany ciąg

  Cel: znajdź wybrany element
- Informatyka: przeszukiwanie binarne, programy (Uwaga: nie taki prosty, można posłużyć się gotowym)
- Wsparcie: zgadywanie liczby, zadania z Bobra (multum)

Programy – gotowe, ale … *show your work*

```
def szukaj_z_in(x, lista):
  if x in lista:
    print("x[",lista.index(x),"] =",x)
  else:
    print("lista nie zawiera elementu", x)
```

```
def szukaj_z_wartownikiem(x, lista):
  lista = lista + [x]
  i = 0
  while lista[i] != x:
    i = i + 1
  if i < len(lista) - 1:
    print("lista[",i, "] =",x)
  else:
    print("lista nie zawiera elementu", x)
```

```
def BinarySearch(y,a,k,l):
  Lewy,Prawy=k,l
  while Lewy <= Prawy:
    Środek=(Lewy+Prawy) // 2
    if a[Środek] == y:
      return Środek
    else:
      if a[Środek] < y:
        Lewy=Środek+1
      else:
        Prawy=Środek-1
  return -1
```

```
def Scal(l1,l2,l3):
 # Scalanie podciągów uporządkowanych
 # l1 i l2 w podciąg uporządkowany l3
  i1,i2,i3=0,0,0
  n1,n2=len(l1),len(l2)
  while i1 < n1 and i2 < n2:
    if l1[i1] < l2[i2]:
      l3[i3],i1=l1[i1],i1+1
    else:
      l3[i3],i2=l2[i2],i2+1
    i3=i3+1
  while i1 < n1:
    l3[i3],i1,i3=l1[i1],i1+1,i
  while i2 <n2:
    l3[i3],i2,i3=l2[i2],i2+1,i
```

```
def Sort(d,g):
  # Rekurencyjne porządkowanie ciągu w liście
  l,p = d,g
  v=x[(d+g) // 2]
   # Podział ciągu elementem środkowym
  while l <= p:
    while x[l] < v:
      l=l+1
    while v < x[p]:
      p=p-1
    if l <= p:
      x[l],x[p] = x[p],x[l]
      l,p = l+1,p-1
  # Porządkowanie podciągów
  if d<p:
    Sort(d,p)
  if l<g:
    Sort(l,g)

def QuickSort(x):
  n=len(x)
  Sort(0,n-1)
```

```
def InsertionSort(x):
  n=len(x)
  x[0]=-1
  for i in range(2,n):
    y=x[i]
    k=i-1
    while y<x[k]:
      k=k-1
    for j in range(i-1,k,-1):
      x[j+1]=x[j]
    x[k+1]=y
```

```
def SelectionSort(lista):
  for i in range(0,len(lista)-1,1):
    k=i;
    for j in range(i+1,len(lista),1):
      if lista[k]>lista[j]:
        k=j
    r=lista[i]
    lista[i]=lista[k]
    lista[k]=r
  for r in lista:
    print(r, end=" ")
```

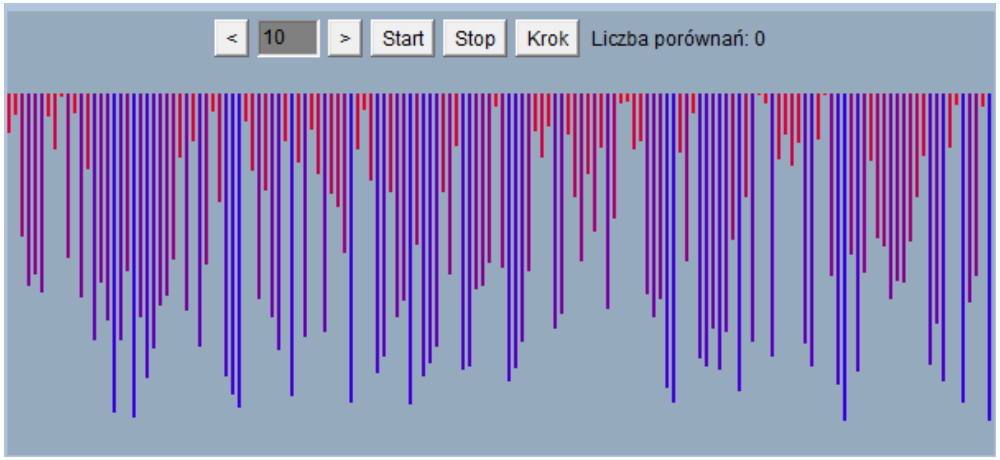
```
def BubbleSort(lista):
    # Porządkowanie metodą bąbelkową
  Kres=len(lista)-1
  while Kres > 0:
    k=0;
    for i in range(0,Kres,1):
      if lista[i]>lista[i+1]:
        r=lista[i]
        lista[i]=lista[i+1]
        lista[i+1]=r
        k=i
    Kres=k
  for r in lista:
    print(r, end=" ")
```

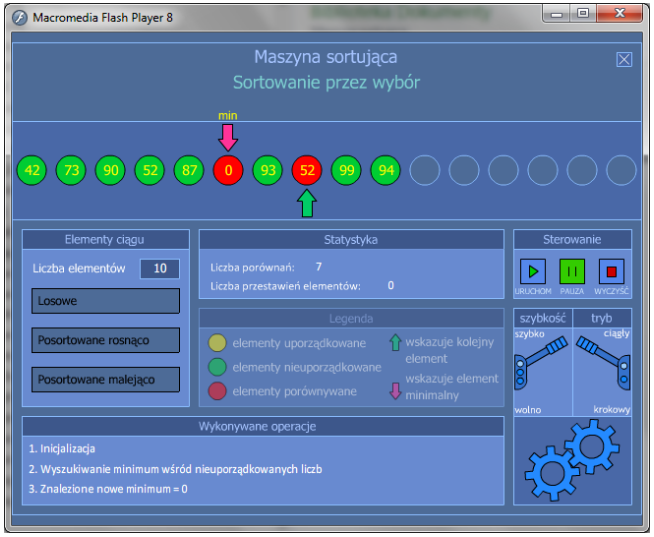
Porządkowanie … przez 12 lat w szkole, 4

4. Gim

- Sytuacja: ciąg liczb

  Cel: uporządkuj
- Informatyka: iteracja: najmniejszy na początek, przestawić elementy w złej kolejności, pierwsze algorytmy porządkowania, pierwsze programy sortujące (Python)
- Wsparcie: programy demo, Godzina Kodowania (programowanie)



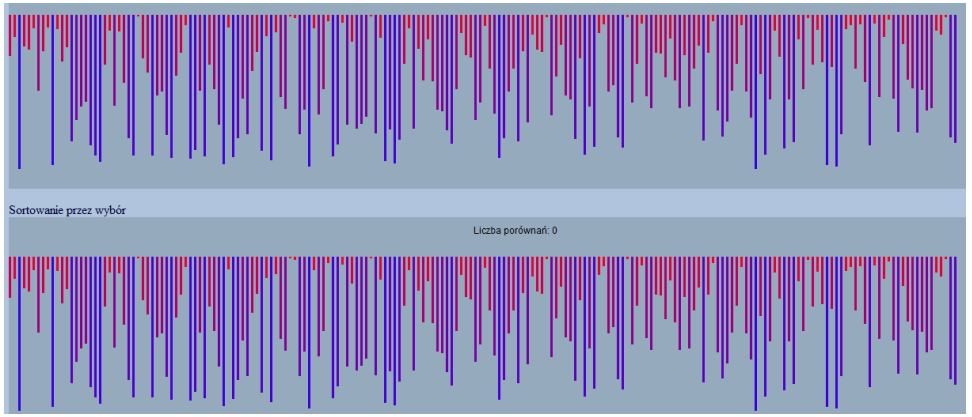


Porządkowanie … przez 12 lat w szkole, 5

4. Gim, LO, LO rozszerzenie

- Sytuacja: ciąg liczb

  Cel: uporządkuj
- Informatyka: elementy komputerowych implementacji: operacja scalania – porządkowanie przez scalanie, rekurencja, programowanie
- Informatyka – pytania ogólne, np.:
  - znaczenie porządku – łatwo znaleźć – szukanie przez podział ciągu – zamiast 1000 prób, tylko 10 – gra w zgadywanie liczby wśród 1000
  - porównanie efektywności metod: przez wybór – stała liczba działań, bąbelkowa – szybka na mało nieuporządkowanym ciągu, porównanie czasów obliczeń



Algorytm Euklidesa – problemy, pojęcia, algorytmy

Tok zajęć 1-3, 4-6:

1. Problem przelewania wody:

- Czy czerpakami 4l i 9l można napełnić naczynie 6l?
- A czerpakami 4l i 6l naczynie 15l?

Zabawa na otwartym powietrzu – podejmowanie prób a później w klasie – rozwiązanie: wzięliśmy 2*9l i wylaliśmy 3*4, czyli:

$9 * 2 – 4 * 3 = 6$ GREAT!

A czy istnieją x i y takie, że: $4 * x + 6 * y = 15$?

2. Wieże z klocków:

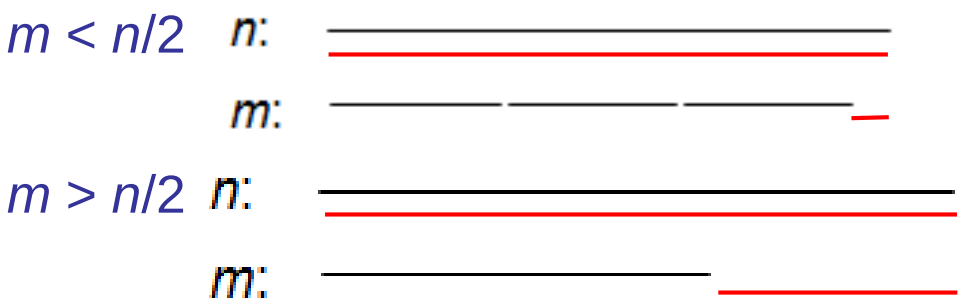
Dwie kupy klocków o jednakowych rozmiarach, np. 4 i 6.

Ustawić możliwie najniższe wieże osobno z jednych i osobno z drugich o tej samej wysokości. Jak?

Myślenie logarytmiczne

logarytm i algorytm to anagramy

- Logarytm ukryty w algorytmach:
- Algorytmiczna definicja logarytmu: ile razy należy podzielić przez dwa liczbę i jej ilorazy, by osiągnąć 1 – można wprowadzić już w gimnazjum!
- Euklides mógł wynaleźć logarytm 300 lat p.n.e., a zrobił to dopiero John Napier 400 lat temu, w 1614 roku.



n	m	r_i
34	**21**	13
21	13	8
13	8	5
8	5	3
5	3	2
3	2	**1**
2	1	0

- Generowane liczby są połowione co druga
- A zatem, dla 10^{300}, algorytm Euklidesa wykonuje ok. 2000 mnożeń – to chwila, chwilę czekamy na zaszyfrowany mail.

Język, języki … - komunikacja z komputerem

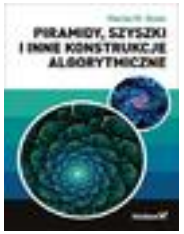
Granice naszego języka programowania technologii

są granicami naszego poznania świata za pomocą technologii

[Maciej M. Sysło]

Tutaj:

Programowanie technologii = kreatywne korzystanie z technologii, ale nie tylko programowanie w języku programowania



Wybór języka

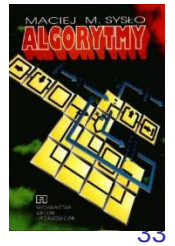
K-3, 4-6 – język obrazkowy, wizualny, blokowy

Scratch, Blockly – zaleta: jest w Godzinie kodowania, w Robotach, Baltie

4-6, gimnazjum, LO – język „tekstowy” – proponuję Python, ale może być C++, raczej już nie Pascal, …

Ważne

Przejście między językami (4-6 – Gim) – te same konstrukcje programistyczne/algorytmiczne



Inicjatywy związane z programowaniem

- Wykład twórcy języka Scratch, Mitchela Resnicka: po angielsku: http://edu.rsei.umk.pl/wcce2013/?q=node/60 z przekładem: http://edu.rsei.umk.pl/iwe2013/?q=node/21
- Godzina kodowania – ponad 150 mln uczestników http://godzinakodowania.pl/ – np. zajęcia dla uczniów, którzy nie czytają
- Wiele innych inicjatyw krajowych i zagranicznych
  - Konkurs Bóbr – http://bobr.edu.pl/
  - Akademia Khana – https://pl.khanacademy.org/
- Inicjatywy komercyjne:
  - Programowanie robotów – Wonder: http://www.wonderpolska.pl/
  - Inicjatywa Samsunga: Mistrzowie kodowania

Myślenie – nie tylko informatyczne

Świetny przykład (A.B. Kwiatkowska): próbujemy dopasować jedno do drugiego, porównać zgodność jednego z drugim, lub tylko z fragmentem:

- W edytorze: wyszukanie frazy w tekście
- DNA – na ile jest ono wspólne dla bliźniąt?
- DNA – czy zawiera pewne fragmenty związane z chorobami dziedzicznymi?
- W jakim stopniu pokrywają się prace – plagiat?
- Informatyka daje metodę/narzędzie dopasowanie wzorca


USB
BLE ANTENNA
RESET
BATTERY
BBC
micro:bit
PROCESSOR
COMPASS
ACCELEROMETER
PINS


0
1
2
3V
GND

Wdrożenie – wsparcie rozwoju nauczycieli

1) standardy przygotowania nauczycieli do prowadzenie zajęć z informatyki na różnych etapach edukacyjnych
2) system ewaluacji pracy nauczycieli informatyki, podczas regularnych zajęć z uczniami w klasie;
3) ramowe programy zajęć w uczelniach:
   - studiów podyplomowych
   - studiów nauczycielskich dla przyszłych nauczycieli informatyki na poszczególnych etapach edukacyjnych
   - modułów informatycznych, do kształcenia w uczelniach przyszłych nauczycieli nauczania początkowego i przedszkolnego
4) programy kursów doskonalących nauczycieli, którzy mają uprawnienia do nauczania informatyki;
5) certyfikat – potwierdzenie przygotowania do prowadzenia zajęć z informatyki

Przygotowanie nauczycieli – standardy

Standardy są w 4 grupach. Nauczyciel:

1. Kompetencje przedmiotowe: Wykazuje się znajomością informatyki w zakresie, w jakim naucza i stosuje tę dziedzinę w szkole, i umiejętnościami wyjaśniania pojęć i zasad tej dziedziny oraz przekazywania ich innym
2. Kompetencje metodyczne: Celowo i efektywnie posługuje się metodami nauczania informatyki
3. Kompetencje technologiczne: Rozwija środowisko kształcenia informatycznego
4. PD – profesjonalny rozwój: Angażuje się w profesjonalny rozwój

Standardy są określone na trzech poziomach:

- zintegrowanym, dla nauczycieli nauczania początkowego
- podstawowym, dla nauczycieli informatyki: w szkołach podstawowych w klasach 4-6, w gimnazjach i w szkołach ponadgimnazjalnych (informatyka na poziomie podstawowym)
- rozszerzonym, dla nauczycieli informatyki w zakresie rozszerzonym

Standardy na danym poziomie edukacyjnym obejmują również standardy na poprzednich poziomach, mają więc charakter przyrostowy

Przygotowanie nauczycieli – standardy – przykłady

Schooling is about student achievement

Standardy	Kryteria osiągnięć (wskaźniki)		
	Poziom zintegrowany	Poziom podstawowy	Poziom rozszerzony
a. aranżuje rzeczywiste sytuacje, które uczniowie abstrahują w postaci danych i powiązań (relacji) między nimi oraz celu do osiągnięcia [*rzeczywiste sytuacje – dane jako abstrakcja*]	• inspiruje uczniów do znajdowania podobnych cech, wzorców i układów w udostępnionych im obiektach, by je grupowali i porządkowali • zachęca uczniów do aktywnego planowania i osiągnięcia celu wynikającego z sytuacji problemowej • podczas powyższych aktywności, zachęca uczniów do posługiwania się obrazkami i modelami, do rozmów i wymiany informacji oraz do poruszania się	• przedstawia uczniom sytuacje problemowe, inspirujące do formułowania specyfikacji problemów, dla których mają zaprojektować algorytmy i napisać programy • angażuje uczniów do celowego i efektywnego przetwarzania danych i informacji, powiązanych z różnymi sytuacjami problemowymi	• angażuje uczniów do analizowania rzeczywistych sytuacji problemowych, jako źródła problemów, rozwiązywanych następnie algorytmicznie za pomocą komputera

Standardy	Kryteria osiągnięć (wskaźniki)		
	Poziom zintegrowany	Poziom podstawowy	Poziom rozszerzony
a. angażuje uczniów do tworzenia algorytmów dla wybranych sytuacji problemowych [*algorytmy dla sytuacji problemowych*]	• inicjuje zabawy uczniów w odtwarzanie ciągu poleceń • zachęca uczniów do prezentowania codziennych czynności w postaci instrukcji (algorytmu) • angażuje uczniów do zespołowego wykonania algorytmu poza komputerem	• angażuje uczniów do formułowania specyfikacji sytuacji problemowej i zaprojektowania dla niej algorytmu	• angażuje uczniów do formułowania specyfikacji złożonej sytuacji problemowej i zaprojektowania lub dobrania dla niej odpowiedniego algorytmu
g. wspiera uczniów w rozwijaniu ich umiejętności programowania z użyciem języka programowania [*programowanie*]	• zachęca uczniów do tworzenia programów według własnych pomysłów w wizualno-blokowym środowisku programowania, zawierających podstawowe konstrukcje, w tym interakcję w odpowiedzi na zdarzenia i zmienne • stwarza uczniom sytuacje do pro-	• na odpowiednio dobranych przykładach angażuje uczniów do pełnej realizacji procesu programowania w wybranym środowisku: projekt programu, umieszczenie programu w systemie, uruchamianie programu, poprawianie błędów (debugowanie), testowanie	• przez dobór odpowiednich problemów skłania uczniów do stosowania w programach złożonych typów danych, w tym dynamicznych, i złożonych konstrukcji programistycznych, takich jak: instrukcje zagnieżdżone, rekurencja • angażuje uczniów do zespołowych

Dziękuję Państwu za uwagę
i proszę nie zapomnieć:

http://mmsyslo.pl