

EDUKACJA INFORMATYCZNA NA TLE TEORII PEDAGOGICZNYCH

Maciej M. Sysło
Warszawska Wyższa Szkoła Informatyki
syslo@ii.uni.wroc.pl; <https://mmsyslo.pl>

Abstract. *Many theories of teaching and learning have been formulated, but there is no single one that could be applied in every educational situation. The appearance of computers and technology in education has led to the development and enrichment of existing theories, as well as the emergence of new ones. In this paper, we refer only to those general issues that were significantly related to the development of computer science education.*

Komputer i towarzyszące mu urządzenia oraz technologia towarzysząca komputerom od samego początku miały znacznie większy wpływ na edukację niż jakiegokolwiek inne pomoce dydaktyczne (naukowe), wcześniejsze jak i obecne w tym samym czasie. W tym rozdziale wybraliśmy najważniejsze idee i teorie pedagogiczne [13, 8] i przedstawiamy zwięźle te ich elementy, które bezpośrednio lub pośrednio można odnieść do wykorzystania komputerów, informatyki oraz technologii w edukacji. Stanowi to tło dla historii edukacji informatycznej w Polsce.

Sformułowano wiele teorii nauczania i uczenia się, ale nie ma jednej, która mogłaby być zastosowana w każdej sytuacji edukacyjnej. Wśród wielu teorii uczenia się, każda może być użyteczna w odpowiednio wybranym kontekście. Pojawienie się w edukacji komputerów i technologii doprowadziło do rozwinięcia i wzbogacenia istniejących teorii, jak i powstanie nowych, które często wykorzystują wcześniejsze koncepcje. Odnosimy się tylko do tych kwestii ogólnych z obszaru pedagogiki, z którymi istotnie wiązał się rozwój edukacji informatycznej, staramy się przy tym zachować porządek chronologiczny dla opisywanych koncepcji i ich twórców.

1. Progresywizm, pragmatyzm: John Dewey (1859-1952)

John Dewey to jeden z najwybitniejszych filozofów, psychologów i pedagogów, współtwórca i propagator **progresywizmu**, znanego także jako Ruch Nowej Edukacji. Jego idee nadal wywierają olbrzymi wpływ na postrzeganie wychowania i nauczania na całym świecie. Oparł je na filozofii **pragmatyzmu**, podkreślając praktyczne konsekwencje wiedzy i nauczania. Za podstawowe źródło wiedzy i jej weryfikacji uznawał naukę

przez doświadczenie, określane jako *uczenie się przez działanie* (ang. *learning by doing*). W szkole nie ma być wydzielonych klas, zajęć uczniów kierowanych przez nauczycieli według sztywnych programów nauczania, przekazywania podręcznikowej wiedzy, gdyż to zabija aktywność uczniów. Uczenie się powinno mieć charakter społeczny i interaktywny i przebiegać w środowisku, w którym uczniowie mogą prowadzić doświadczenia i wpływać na to, czego się mają uczyć, uczestnicząc w ten sposób we własnej nauce. Koncepcję edukacji dla demokracji [4] Dewey oparł na wolności uczniów i uszanowaniu ich zainteresowań oraz na wspólnym działaniu uznając je za najlepszy sposób tworzenia wiedzy, rozwijania umiejętności i poszerzania kompetencji. Celem edukacji nie powinno być nabywanie z góry określonego zestawu umiejętności, ale aktywna realizacja potencjału jednostki w kształtowaniu umiejętności przez doświadczenie i rozwiązywaniu problemów.

Uznaje się również Deweya za ojca **podejścia problemowego** (ang. *Problem-Based Learning*, PBL), jako metody aktywnego dociekania, szeroko stosowanej dzisiaj w wielu obszarach edukacji, m.in. w matematyce (G. Polya) i informatyce. Za problem uznawał sytuację, którą uczeń lub grupa uczniów napotyka w swoim otoczeniu, a jej rozwiązanie ma im dostarczyć odpowiedniej wiedzy i wykształcić pewne umiejętności. Zadaniem szkoły jest stwarzanie takich sytuacji uczniom. Sugerowane przez Deweya etapy radzenia sobie z sytuacjami problemowymi nieco różnią się od etapów zalecanych dzisiaj w przypadku rozwiązywania problemów matematycznych i informatycznych, co wynika z charakteru sytuacji problemowych, które miały być przedmiotem aktywności uczniów. Widział on następujące etapy postępowania:

- zauważenie i odczucie trudności w otoczeniu lub w pewnej sytuacji;
- zgłębienie nurtującej trudności prowadzące do określenia problemu, który ma być rozwiązany;
- poszukanie i zaproponowanie rozwiązania określonego problemu;
- weryfikacja rozwiązania na drodze rozumowania i analizy wniosków wypływających z rozwiązania;
- empiryczna weryfikacja rozwiązania na drodze do jego przyjęcia/ odrzucenia.

System edukacji w Polsce, ale nie tylko w naszym kraju, nawet nie przymierza się do idei Deweya, by indywidualizować proces uczenia się na przykład przez osłabienie klasowo lekcyjnego trybu zajęć szkolnych, podczas którego z założenia na ogół wszyscy uczniowie uczą się tego samego i w tym samym tempie.

Zajęcia informatyczne w znacznym stopniu umożliwiają uczniom własny wybór zagadnień i metod, tempa ich realizacji oraz finalnych rozwiązań według własnych pomysłów i indywidualnych możliwości ich realizacji. Ponadto, w podstawie programowej informatyki w odniesieniu do najmłodszych uczniów zapisano m.in., że „uczeń [...] formułuje i zapisuje w postaci algorytmów polecenia składające się na: a) rozwiązywanie problemów z życia codziennego i z różnych przedmiotów”.

2. Behawioryzm, nauczanie programowane: Burrhus F. Skinner (1904-1990)

Burrhus F. Skinner to czołowy przedstawiciel **behawioryzmu**. W myśl tego nurtu psychologii i pedagogiki, uczenie się jest wynikiem oddziaływania środowiska na uczącego się i objawia się ono zmianą jego zachowania (np. ucząc się informatyki uczeń potrafi formułować algorytmy). Większość badaczy behawioryzmu opierała swoje konkluzje na wynikach eksperymentów ze zwierzętami¹, które potwierdzały, że powodzenie w wielokrotnym działaniu poprawia sposób działania (czyli efekt uczenia się) a czekająca nagroda zwiększa chęć podejmowania działań. W swoich badaniach B.F. Skinner zauważył jednak, że uczeniu się zwierząt nie musi zawsze towarzyszyć nagroda, może pojawiać się przypadkowo. Podobnie w przypadku uczniów – pochwały i drobne wyróżnienia ale od czasu do czasu wyraźnie mobilizują uczniów, wzmacniają (ang. *reinforce*) ich reakcje (zachowanie).

W interpretacji uczenia się przez behawiorystów, nauczyciel pełni „kierowniczą rolę” w przygotowaniu i prowadzeniu zajęć określając ich cele (w tym metody i narzędzia) oraz spodziewane (czyli obserwowane) zachowania uczniów, bez konieczności zainteresowania się ich nabytą wiedzą.

B.F. Skinner wyróżniał się wśród przedstawicieli behawioryzmu badaniami nad maszynami do nauczania [15]. Swoje zainteresowania w tym kierunku uzasadniał także argumentami społecznymi. Twierdził, że „Na świecie jest więcej ludzi niż kiedykolwiek wcześniej (1958), a większa ich część chce edukacji. Popyt nie może zostać zaspokojony po prostu przez budowę większej liczby szkół i szkolenie większej liczby nauczycieli. Edukacja musi stać się bardziej wydajna. W tym celu programy nauczania muszą zostać zrewidowane i uproszczone, a podręczniki i techniki nauczania ulepszone.” Co więcej „Uczeń staje się coraz bardziej biernym odbiorcą instrukcji. Z eksperymentalnych badań nad uczeniem się wywodzą się urządzenia, które tworzą optymalne warunki do samokształcenia.” Tak zrodziła się idea nauczania programowanego [14].

Nauczanie programowane

Wzrost możliwości komputerów w latach 1950-1960 przyczynił się do wzrostu popularności **nauczania programowanego**. Tę koncepcję dydaktyki instrukcyjnej można przyrównać do idei „programowanie ucznia” (niekoniecznie za pomocą komputera), polegającej na niemal automatycznym prowadzeniu go przez kolejne etapy zdobywania wiedzy według przyjętego „programu”. Nauczanie programowane polega na trzech podstawowych zasadach [10]:

- (1) podziału materiału na małe, logicznie uporządkowane kroki;

¹ Jest to źródłem zarzutów stawianych behawiorystom, że pomijali sferę psychiczną uczenia, a obserwowali jedynie zachowania (reakcje zwierząt), w schemacie bodziec – reakcja.

- (2) natychmiastowej oceny każdej odpowiedzi;
- (3) indywidualizacji tempa i treści nauczania.

Do tego znakomicie może nadawać się komputer. Programowanie ucznia może stwarzać wrażenie indywidualizacji, gdyż każdego ucznia można inaczej zaprogramować, różnymi porcjami materiału i w odpowiednim dla niego tempie, programować. Ponadto, komputer może przechowywać wiele danych i analizować je tworząc sylwetkę ucznia w jaki sposób się uczy. Jednak jest to nadal kierowanie uczniem przez komputer, w sposób, w jaki zaprogramowana została maszyna, a uczeń może poruszać się jedynie w ramach opcji przewidzianych w programie i nie może podążać w pełni własną drogą. Wykorzystanie komputerów w nauczaniu programowanym było w pewnym sensie unowocześnieniem tego sposobu kształcenia. Ta chęć programowania ucznia jest nadal bardzo popularna, a nowe aplikacje informatyczne tylko uatrakcyjnijają to podejście.

System PLATO

W dużych i potężnych, jak na tamte czasy maszynach, upatrywano narzędzia do realizacji nauczania programowanego. Komputery były jednak wtedy instalowane głównie w specjalnych ośrodkach (w tym na większości uniwersytetów w USA), a dostęp do nich mieli tylko informatycy. Należało więc podjąć działania, które umożliwią dostęp do komputera większej liczbie uczących się użytkowników, nie koniecznie informatyków, a przede wszystkim stworzyć odpowiednie oprogramowanie. Tak zrodził się system PLATO (ang. *Programmed Logic for Automated Teaching Operations* – Programowana Logika dla Automatycznej Operacji Nauczania), będący przykładem komputerowego wspomaganie nauczania (ang. *Computer assisted Instruction* – **CAI**). Można uznać, że system PLATO to *Personal Computing before Personal Computers*.

Generalnie, na nauczanie wspomagane komputerem – CAI – składa się:

1. Przygotowanie przez nauczycieli materiałów pod względem merytorycznym i pedagogicznym, jednolitych, co do formy i treści i zgodnych z wybranymi programami nauczania.
2. Indywidualizacja kształcenia, ograniczona do zakresu oferowanego przez system.
3. Bieżąca, okresowa i finalna ocena postępów uczącego się.
4. Dodatkowo, CAI może służyć do wychwytywania uczniów uzdolnionych lub z trudnościami w uczeniu się i traktowania ich indywidualnie.

Rozwój systemu PLATO przez ponad 40 lat może być przykładem wpływu koncepcji nauczania i uczenia się oraz nowych technologii na środowiska komputerowego wspomaganie nauczania wzbogacone technologią.

W systemie PLATO zrealizowano wiele innowacyjnych rozwiązań, które zmodyfikowane do aktualnie dostępnej technologii spotykamy we współczesnych rozwiązaniach adresowanych do edukacji (nie tylko). Przede wszystkim opracowano wiele koncepcji umożliwiających wielodostęp, jak: fora dyskusyjne, tablice ogłoszeń, testy online, pocztę, pokoje czatów, komunikatory internetowe, zdalne udostępnianie ekranu i gry dla

wielu graczy. Pod nadzorem systemu PLATO rodziły się społeczności uczących się. System PLATO od wersji IV oferował tekst, grafikę i animację niezbędne do tworzenia materiałów szkoleniowych, dysponował również pamięcią wspólną, umożliwiającą interakcję między użytkownikami.

System PLATO oferował również wiele funkcji przydatnych przy organizacji procesu nauczania i uczenia się, jak m.in. kontekstową ocenę odpowiedzi formułowanej w postaci swobodnego tekstu oraz informacje zwrotne zaprojektowane w celu adekwatnego reagowania na alternatywne odpowiedzi.

Brak jest informacji, by system PLATO był stosowany w Polsce. Prowadzono natomiast prace nad innymi systemami CAI. Jednym z nich był system OSKAR rozwijany w latach 1978-1980 przez zespół w Akademii Ekonomicznej (obecnie Uniwersytet Ekonomiczny) we Wrocławiu pod kierunkiem Jerzego Korczaka [9].

Obecnie, reakcja środowisk edukacyjnych na system PLATO i inne typu CAI jest mieszana. Z jednej strony, takie systemy mogą prowadzić do dehumanizacji kształcenia przez eliminację nauczyciela. Z drugiej zaś, zaprojektowane, by uwzględniać różne drogi rozwoju uczniów, tzw. **systemy adaptacyjne**, mogą być wsparciem zarówno dla samych uczniów, jak i w procesie nauczania kierowanym przez nauczyciela.

3. Konstruktywizm: Jean Piaget, Lew S. Wygotski, Jerome S. Bruner

Wymienieni w tytule tego punktu badacze to prekursorzy **konstruktywizmu**, nurtu w psychologii poznawczej, krytycznego wobec behawioryzmu. Konstruktywizm zastępuje wizję ucznia biernie odbierającego bodźce docierające do niego z zewnątrz i reagującego zachowaniem, konstruowaniem przez niego struktur poznawczych będących umysłowymi reprezentacjami poznawanej rzeczywistości. W tej filozofii poznania i nauczania, uczenie się przebiega przez interakcję z otoczeniem i eksperymentowanie. Wiedza zaś jest konstruowana przez ucznia w powiązaniu z już posiadaną i stale się rozwija na bazie nowych doświadczeń. Chociaż konstruktywizm zaleca skupienie uwagi na podejściu indywidualnym do uczniów, ważne jest przy tym uczenie się w grupie i interakcja między uczniami. Nauczyciel powinien wspierać uczniów zadaniami i pytaniami, skłaniającymi ich do refleksji i dalszych dociekań. Konstruktywizm jest więc bardziej teorią uczenia się niż nauczania.

Rozwinęło się wiele koncepcji i wariantów konstruktywizmu. Dalej skupimy uwagę na dwóch z nich: konstruktywizmie rozwojowo-poznawczym Jeana Piageta i konstruktywizmie społeczno-kulturowym Lwa S. Wygotskiego i Jerome'a S. Brunera. Obie przyjmują koncepcję ucznia, jako osoby aktywnie uczącej się, konstruującej własną wiedzę jako obraz rzeczywistości, w której przychodzi mu żyć. W dalszym punkcie odniesiemy się jeszcze do wkładu Seymoura Paperta, pioniera komputerów w edukacji, do rozwoju **konstrukcjonizmu**.

Jean Piaget (1896-1980)

Jean Piaget, przedstawiciel konstruktywizmu rozwojowo-poznawczego, wspierał swoje koncepcje szerokimi badaniami, polegającymi na rozmowach klinicznych z dziećmi. Badania te uzasadniały, że indywidualne działania dziecka są głównym motorem jego rozwoju umysłowego, uczenie się u dziecka zachodzi więc poza bezpośrednimi wpływami otoczenia i osób dorosłych. Wadą tego podejścia było uznanie, że uczenie się jest zjawiskiem indywidualnym, chociaż wszyscy żyjemy wśród ludzi, Piaget nie doceniał środowiska społecznego.

J. Piaget wyróżnił 4 etapy dojrzewania struktur poznawczych u dzieci:

- sensomotoryczny (1-2 roku życia), tworzenie początkowej struktury poznawczej
- przedoperacyjny (2-7 roku życia), operowanie w konkretnych sytuacjach fizycznych (np. operacje arytmetyczne na przedmiotach);
- operacji konkretnych (7-12 rok życia), konstruowanie struktur logicznych (np. operacje arytmetyczne na liczbach);
- operacji formalnych, opanowanie rozumowania abstrakcyjnego.

Ten ostatni etap trwa od około 12. roku życia do dorosłości i według Piageta dopiero na tym etapie człowiek wykazuje zdolność do myślenia abstrakcyjnego i do używania symboli związanych z abstrakcyjnymi koncepcjami. Ta interpretacja rozwoju ucznia stwarza kłopoty z zaplanowaniem miejsca – najlepiej od najmłodszych lat – w kształceniu informatycznym dla rozwijania umiejętności uczniów w zakresie programowania, które wymaga posługiwania się abstrakcyjnym myśleniem². Poniżej, w opisie rozwoju koncepcji J.S. Brunera pokazujemy, jak uporać się z tą trudnością.

W jednej ze swoich prac³ J. Piaget dotknął kwestii, z którą parają się wszyscy nauczyciele, a mianowicie: jeśli uczeń wykonał zadanie (np. napisał program i przetestował jego działanie), to czy zrozumiał, co robił i co zrobił, czy zrozumiał proces wytwarzania, czyli czego się nauczył i jaką skonstruował wiedzę, i dodatkowo, jak myślał? Krótko – czy wykonanie projektu może być potwierdzeniem tego, że zaszło uczenie się? J. Piaget pisze o niezbędnej zdolności w tym przypadku do konceptualizacji działania i refleksji nad sukcesem, o potrzebie **uchwycenia (zrozumienia) świadomości**, czyli refleksji nad abstrakcją.

Lew S. Wygotski (1896-1934)

Lew S. Wygotski to przedstawiciel konstruktywizmu społeczno-kulturowego, który w swoich badaniach i poglądach uwzględniał, że uczenie się przebiega w środowisku kulturowo-społecznym, w którym dzieci konstruują swoją wiedzę w interakcji z innymi dziećmi i dorosłymi. Klasyfikację poziomów umysłowego rozwoju dzieci w sensie

² W pracy [20] cytujemy opinie na ten temat m.in. Niklausa Wirtha i Edsgera W. Dijkstry.

³ Bardziej szczegółowo piszemy o tym w pracy [21].

J. Piageta traktował jako statyczne przyporządkowanie i przyjmował, że dzieci nawet na tym samym poziomie rozwojowym odpowiednio pobudzone i stymulowane mogą rozwijać się inaczej. Wyróżniał Strefę Najbliższego Rozwoju dziecka, w której znajdują się zadania nieco trudniejsze, ale wykonalne przez ucznia z pomocą osoby doświadczonej – to wsparcie J.S. Bruner nazwał „rusztowaniem” – oraz Strefę Aktualnego Rozwoju, w której uczeń jest w stanie samodzielnie wykonać stawiane mu zadania..

Jerome Seymour Bruner (1915-2016)

Jerome S. Bruner to jeden z największych i najbardziej wpływowych przedstawicieli konstruktywizmu, który z kolei uznawał, że proces poznawczy jest uzależniony od kontekstu kulturowego. W nauczaniu zaś uważał, że powinno być dostosowane do etapu rozwoju poznawczego ucznia i uwzględniać jego wcześniejsze doświadczenia i konstrukcje wiedzy. Zaproponował [3] inną, w odróżnieniu od J. Piageta, teorię rozwoju poznawczego, która uznaje prezentację i rozwój tych samych pojęć na różnych etapach⁴. Wyróżnił trzy etapy rozwoju określane sposobami reprezentacji:

- enaktywny (bazujący na manipulowaniu i interakcji z rzeczywistymi obiektami – 0-2 rok życia);
- ikoniczny (oparty na manipulowaniu obrazami (reprezentacja) obiektów lub zjawisk – od 2 do 6 lat);
- symboliczny (oparty na manipulowaniu abstrakcyjnymi (słownymi) reprezentacjami rzeczywistych obiektów i zjawisk – powyżej 6 lat).

Chociaż przechodzenie na poszczególne etapy rozwoju jest, podobnie jak u Piageta, związane z wiekiem, to jednak w odróżnieniu od hierarchicznego porządku Piageta, Bruner twierdził, że wszystkie trzy etapy w jego teorii mogą współistnieć i każdy człowiek używa tych trzech trybów reprezentacji od czasu do czasu, a dziecko najlepiej uczy się przechodząc, najlepiej samodzielnie, od konkretnych doświadczeń (manipulowanie przedmiotami) do reprezentacji wizualnych (świat w obrazach), a następnie do abstrakcyjnego myślenia (język, słowa, symbole, pojęcia).

W jaki sposób teorie J. Piageta i J.S. Brunera odzwierciedlają naukę programowania w szkole i mogą stanowić bazę teoretyczną dla jej praktycznego kształtowania i rozwoju? Ponieważ programowanie jest związane z umiejętnością abstrakcyjnego myślenia, wymaga więc rozwoju poznawczego uczniów na wyższym etapie. Dlatego z perspektywy najmłodszych oraz nieco starszych uczniów, dbając o ich spiralny rozwój, powinno się udostępniać im środowiska programistyczne, które umożliwiają obniżanie poziomu abstrakcji podstawowych koncepcji programowania i związanych z nimi pojęć informatycznych. W konsekwencji wybrane środowiska programistyczne powinny wpisywać się w jeden lub więcej etapów poznawczego rozwoju uczniów. Takie właściwości

⁴ Ta cecha teorii rozwoju poznawczego J.S. Brunera stanowi bazę dla podstaw programowych i programów nauczania przyjmujących jako zasadę spiralny rozwój poznawczy uczniów.

mają środowiska programowania blokowego, jak code.org, Scratch i wiele innych, które pasują do etapu enaktywnego (uczniowie klikają i przenoszą obiekty na planszy i ekranie) i ikonicznego (rozwiązują łamigłówki i kodują je). Środowisko języka Python, do którego uczniowie przechodzą w starszych klasach, pasuje do symbolicznego etapu poznania, na którym manipulują abstrakcyjnymi reprezentacjami obiektów i akcji pochodzących z rzeczywistych sytuacji problemowych.

Jedną z edukacyjnych propozycji J.S. Brunera (1960) było **uczenie się przez odkrywanie** lub dociekanie (ang. *discovery* lub *inquiry-based learning*), proponowane wcześniej przez J. Deweya, a później – przez S. Paperta. Związane jest ono z eksperymentowaniem i aktywnym poszukiwaniem w przeciwieństwie do biernej asymilacji wiedzy, a w przypadku J.S. Brunera dodatkowo – przebiega ono podanymi przez niego etapami. Nie tylko w tym podejściu, J.S. Bruner upatrywał specjalną rolę nauczyciela, jako przewodnika (tutora, **facilitatora**), zachęcającego uczniów do samodzielnego budowania swojej wiedzy, zwłaszcza w procesie spiralnego uczenia się. Posłużył się przy tym terminem **rusztowanie** (ang. *scaffolding*) na opisanie procesu tymczasowego wsparcia uczącego się przez nauczyciela, stopniowo ustępującego wraz z postępami uczącego się.

J.S. Bruner jest też uważany za ojca **spiralnego** (programu) **nauczania**, którego celem jest stopniowy rozwój wiedzy uczniów. Może to dotyczyć konkretnych pojęć lub metod rozwijanych i stosowanych w coraz bardziej rozbudowanych kontekstach, a także całych przedmiotów. Taka jest konstrukcja podstawy programowej informatyki, obowiązującej od 2017/2019. W tym kontekście znany jest jego pogląd, że każdego można wszystkiego nauczyć to zależy tylko od sposobu podania (ang. *any subject could be taught to any child at any age in some form that is honest*). Mają temu służyć wszystkie wymienione wyżej jego koncepcje: (1) trzy etapy rozwoju poznawczego, (2) stosowanie ich w uczeniu się przez odkrywanie i (3) wsparcie tutora (facilitatora) (4) oferującego uczniom rusztowanie.

4. Konstrukcjonizm: Seymour Papert (1928-2016)

S. Papert jest uznawany za pioniera wykorzystania komputerów w edukacji. Prześnięty ideami konstruktywistycznymi po latach spędzonych u Jeana Piageta uważał, że uczeń, jako użytkownik systemu typu CAI, takiego jak PLATO, zachowuje się biernie wykonując głównie polecenia komputera. Odwrócił więc relację i pisał w swoich *Burzach Mózgów* [11]: *Można by sądzić, że komputer jest wykorzystywany do programowania dziecka. W mojej wizji to dziecko programuje komputer*. Papert widział w programowaniu sposób na porozumiewanie się dziecka z komputerem w języku, który rozumieją obie strony. Stworzył w tym celu język Logo, jako pomost między światem uczniów a komputerami.

S. Papert wierzył wraz J. Deweyem i J. Piagetem, że: *dzieci uczą się w działaniu i myśląc o tym, co robią* (ang. *children learn by doing and by thinking about what they*

do) [12]. Te jego słowa z 1970 roku można uznać za zapowiedź poszerzenia konstrukttywizmu do konstrukcjonizmu i uznania myślenia komputacyjnego za nieodłączny element procesu rozwiązywania problemów. Na początku lat 1990' sprecyzował, że **konstrukcjonizm** postrzega uczenie się jako rekonstrukcję niż transmisję wiedzy i najlepsze efekty uczenia się osiąga się przez tworzenie rzeczy, które są namacalne i można je udostępniać. W edukacji informatycznej mogą nimi być np. programy komputerowe, roboty. Później opisał 8 głównych zasad (ang. *Big Ideas*) konstrukcjonistycznego uczenia się [22]:

1. uczenie się przez działanie
2. technologia jako tworzywo
3. zabawa z wyzwaniami w tle
4. uczenie się, jak się uczyć
5. odpowiedni czas na działanie
6. nie ma sukcesu bez niepowodzenia
7. sprawdź, zanim dasz uczniom zadanie
8. jesteśmy w cyfrowym świecie

5. Kognitywizm

Zajmuje się badaniami procesów poznawczych, takich jak myślenie i uczenie się zakładając przy tym, że umysł ludzki przetwarza informacje z otoczenia, podobnie jak komputer. Analizuje więc procesy towarzyszące konstruowaniu wiedzy w umyśle aktywnie uczącego się. W tym sensie, konstruktywizm i konstrukcjonizm są jednymi z nurtów kognitywizmu. Niestety, w rodzimej literaturze przedmiotu [17], technologia informacyjna i informatyka w edukacji są postrzegane głównie jako „obsługa komputera” i nie przekonuje misja tej sfery edukacji:

celem edukacji informatycznej nie jest prezentacja
możliwości komputerów i ich oprogramowania,
ale ujawnianie możliwości uczniów

6. Biegłość w zakresie technologii informacyjnej (ang. Fluency with IT)

Termin **alfabetyzacja informacyjna** (ang. *information literacy*) był stosowany np. w bibliotekoznawstwie na długo przed pojawieniem się komputerów i rozpowszechnieniem technologii informacyjnej i obejmował kompetencje celowego wyszukiwania, oceny i wykorzystywania informacji. Gdy możliwa stała się praca z informacją ze wsparciem technologii cyfrowych (komputerów i Internetu), zaczęto stosować termin **technologia informacyjna** (ang. *information technology*). Wraz z rozpowszechnieniem się tego terminu pojawiła się konieczność określenia, na czym ma polegać alfabetyzacja w zakresie technologii informacyjnej, czyli co każdy musi wiedzieć i rozumieć na temat

technologii informacyjnej, by uczestniczyć w społeczeństwie informacyjnym. Podjął się tego zespół utworzony na dość wysokim szczeblu NSF i NRC w USA⁵. Zaczęto od zdefiniowania technologii informacyjnej, która objęła tradycyjne elementy, jak urządzenia obliczeniowe ogólnego przeznaczenia, urządzenia peryferyjne, systemy operacyjne, aplikacje i informacje, a także urządzenia wbudowane, komunikację i naukę leżącą u podstaw technologii (w tym elementy algorytmiki i programowania). Uwzględniając nie tylko znajomość technologii, ale wiedzę, rozumienie i zdolności intelektualne, postanowiono użyć terminu **biegłość** (ang. *fluency*), który oddaje wyższy stopień kompetencji. Biegłość oznacza tu zdolność korzystania z posiadanej wiedzy, wyrażania siebie w sposób kreatywny oraz tworzenie informacji (a nie tylko jej rozumienia i przetwarzania).

W powyższym sensie, w pierwszej dekadzie tego wieku w szkołach ponadgimnazjalnych był realizowany przedmiot technologia informacyjna.

7. Informatyka bez komputera (ang. Unplugged): Mike Fellows, Tim Bell

Informatykę bez komputera (ang. *Computer Science Unplugged*) definiuje się jako: „zbiór działań i pomysłów angażujących różnych odbiorców pomysłami z informatyki, bez konieczności nauki programowania, a nawet korzystania z urządzenia cyfrowego”. To podejście zostało zainicjowane w 1990 roku przez M. Fellowsa i T. Bella [7] jako program dotarcia (ang. *outreach*) do uczniów w szkołach, które nie mają komputerów i nie uczą informatyki oraz zaangażowania ich „aby pomóc im zrozumieć, czym może zajmować się informatyka poza programowaniem”. Z pedagogicznego punktu widzenia podejście bez komputera opiera się na konstruktywizmie, a częściowo na konstrukcjonizmie. Uczniowie konstruują własną wiedzę, czasami wytwarzając również pewne artefakty, wykorzystując przy tym to, czego się już nauczyli, używając pewnych metod rozumowania i angażując się w sytuacje problemowe do rozwiązania lub pytania, na które należy odpowiedzieć. Ten proces uczenia się prowadzi ich do zrozumienia ważnych pojęć, zasad i mechanizmów o charakterze informatycznym.

Byliśmy w Polsce w dość szczęśliwej sytuacji, bo nawet pierwsze regularne lekcje informatyczne w połowie lat 60., chociaż w znacznej części odbywały się bez komputera, gdy uczniowie pisali programy w klasie, ale jednak później uruchamiali je na komputerze. Również nauczyciele wyjaśniali pojęcia i algorytmy przy użyciu tradycyjnej kredy i tablicy. To podejście pozostało i jest zalecane do dzisiaj, tym bardziej w starszych klasach, gdy pojawiają się trudniejsze problemy i przed posłużeniem się komputerem uczeń powinien zbudować odpowiednią wiedzę, by poradzić sobie z nowymi wyzwaniami.

⁵ Szczegółową relację z prac tego zespołu zawiera raport [1].

Trudno jest dziś utrzymać podejście bez komputera w czystej postaci, gdy prawie wszystkie szkoły są dość dobrze wyposażone w: komputery, tablety, roboty, Internet i lekcje informatyki odbywają się w pracowniach komputerowych. Ponadto, w wielu przypadkach nowych pojęć i metod informatycznych, jak i rozwiązywanych problemów uczeń powinien najpierw skonstruować odpowiednią wiedzę posługując się przy tym myśleniem komputacyjnym, zanim posłuży się komputerem, często stojącym przed nim lub obok. Bywają też pojęcia i problemy, których przybliżenie i wprowadzenie może odbyć się z pomocą komputera i jego aplikacji. Biorąc to pod uwagę proponujemy poszerzenie zakresu stosowania podejścia bez komputera o środowiska nauczania i uczenia się **z komputerem w tle**, w którym komputer (i inne technologie cyfrowe) znajduje się w tle działań edukacyjnych, bliżej lub dalej, mniej lub bardziej zintegrowany, ale nie występuje jako technologia wykorzystywana w nauce programowania.

Prawie każde pojęcie, koncepcję lub metodę informatyczną można przedstawić uczniom bez użycia komputera. Stawiamy jednak na rygorystyczne kształcenie informatyczne, dlatego proponujemy, by podejście z komputerem w tle stosować elastycznie. Zwykle to nauczyciel decyduje o roli komputerów na swoich zajęciach, może jednak pozostawić uczniom swobodę wyboru, by mogli rozwijać także umiejętności podejmowania decyzji o wykorzystaniu i roli komputera i/lub innych technologii w procesie rozwiązywania problemów i uczenia się. Wyróżniamy cztery rodzaje środowisk, w których komputer w pewnym sensie ma swoje miejsce w tle⁶:

- klasyczne środowiska bez komputera, ewentualnie z lamigłówkami komputerowymi
- zadania z konkursu „Bóbr”
- robotyka edukacyjna
- algorytmika i programowanie bez komputera

8. Konektywizm: George Siemens, Stephen Downes

Na początku lat 1990' do szkół zawitał Internet i rozpoczęła się rewolucja w nauczaniu i uczeniu się – opuściły tradycyjne klasy i lekcje. Na początku tego wieku G. Siemens [16] ogłosił powstanie **konektywizmu**, nowej teorii uczenia się w epoce cyfrowej (ang. *A Learning Theory for the Digital Age*), będącej społecznym uczeniem się w sieci.

Konektywizm można uznać za wynik konstatacji nad rolą współczesnych technologii w procesie poznania. Obok tradycyjnych (tj. nie elektronicznych) dróg i zasobów, uczący się ma dzisiaj do dyspozycji sieci połączeń między różnymi źródłami informacji, którymi mogą być zarówno miejsca w sieci, jak i inni użytkownicy sieci. Uczenie się w takich warunkach zostaje wzbogacone o nawigowanie między różnymi źródłami. Wiedza nie jest już tylko efektem własnej konstrukcji we własnym umyśle uczącego się ale można

⁶ Szczegółowe rozważania kontynuujemy w pracy [18].

ją znaleźć w zewnętrznych źródłach. Przy tym, struktura węzłów oraz połączeń i relacji między nimi w sieci ulega ciągłym zmianom.

Konektywizm różni się od innych teorii uczenia się skupiając uwagę na technologii, a przede wszystkim przyjmując, że wiedza może znajdować się poza umysłem uczącego się. Elementy konektywizmu zaczęły pojawiać się w edukacji od momentu dotarcia Internetu do szkół, a zwłaszcza do komputerów uczniów, którzy dość często hołdują uproszczonej wersji konektywizmu – wiedzieć, gdzie – „po co mam się czegoś uczyć (pamiętać), znajdę to przecież lub zapytam kogoś w sieci”.

9. Myślenie komputacyjne: Seymour Papert, Jeanette Wing

W swoim epokowym dziele *Burze mózgów* już w 1980 Seymour Papert odwołał się do **myślenia komputacyjnego** (ang. *computational thinking*), ale dopiero praca Jeanette M. Wing z 2006 roku na ten temat [23] spowodowała uważniejsze przyjrzenie się, na czym powinno polegać rozwiązywanie problemów i uczenie się w dobie komputerów i technologii. Papert ujął to jeszcze w 1980 roku w pełni holistycznie:

chciałbym pokazać, że obecność komputera mogłaby wspierać procesy umysłowe nie tylko instrumentalnie, ale w bardzo zasadniczy, konceptualny sposób, wpływając na to, jak ludzie myślą, nawet wtedy, gdy nie mają fizycznego kontaktu z komputerem.

Z kolei J.M. Wing określiła tym terminem użyteczne postawy i umiejętności, jakie każdy, nie tylko informatyk, powinien starać się wykształcić i stosować. Obecnie przyjmuje się, że:

myślenie komputacyjne to procesy myślowe angażowane w formułowanie problemu i przedstawianie jego rozwiązań w taki sposób, aby komputer – człowiek lub maszyna – mógł skutecznie wykonać.

Przyglądając się bliżej stosowaniu tego podejścia [19], można w nim wyróżnić wiele **sposobów rozumowania** (ang. *mental tools*), które są angażowane zarówno w formułowanie problemu, jak i podczas jego rozwiązywania aż po programowanie rozwiązań. Można więc zalecić zwłaszcza uczniom: *pomyśl komputacyjnie, zanim zaczniesz programować*. J.M. Wing wymieniła w swojej pracy spectrum metod i podejść do modelowania i rozwiązywania problemów zaliczając je do myślenia komputacyjnego, wśród nich: rozpoznawanie wzorców, abstrakcję, redukcję i dekompozycję, aproksymację, rekurencję, podejście heurystyczne.

Innym podejściem do uściślenia znaczenia myślenia komputacyjnego jest operacyjna definicja (CSTA), która porządkuje aktywności uczniów i działań nauczycieli podczas rozwiązywania problemów z różnych dziedzin. Składają się na nią: (1) sformułowanie problemu w postaci, która umożliwia posłużenie się do jego rozwiązania metodami informatycznymi i komputerem, (2) logiczna organizacja danych i wyciągnięcie z nich wniosków, (3) wyabstrahowanie reprezentacji danych w postaci modelu lub

symulacji, (4) rozwiązanie problemu w postaci ciągu kroków, (5) analiza i komputerowa realizacja rozwiązania problemu, (6) przeniesienie doświadczenia nabytego podczas rozwiązywania jednego problemu na inne sytuacje problemowe.

Dzięki tak szerokiemu spojrzeniu na kompetencje informatyczne, informatyka nie jest ograniczana do nauki o komputerach, ale dostarcza metod dla działalności umysłowej, które mogą być wykorzystane z korzyścią dla innych dziedzin, jak i w codziennym życiu. Z drugiej jednak strony, ponieważ praca J.M. Wing ukazała się w czasopiśmie informatycznym, idea myślenia komputacyjnego, adresowana do każdego uczącego się i osób poza systemem edukacji z trudem przebija się do kręgów edukacyjnych w dziedzinach innych niż związane z informatyką. Oczekuje się jednak, że rolę technologii informacyjnej, czyli posługiwania się gotowymi rozwiązaniami komputerowymi, jako informatyki dla wszystkich z przełomu wieków XX i XXI, przejmują obecnie myślenie komputacyjne, znacznie poszerzając obszar i możliwości sposobów i metod rozwiązywania problemów bazujących na ideach wywodzących się z informatyki i jej zastosowań. Przebija się do podstaw programowych kształcenia ogólnego, jednak nadal pozostaje olbrzymim wyzwaniem przekonanie dydaktyków i nauczycieli innych przedmiotów do roli myślenia komputacyjnego w ich dziedzinach.

10. Odwrócone kształcenie: Jonathan Bergman, Aaron Sams

Odwrócone kształcenie się (ang. *flipped learning*) czy też **odwrócona klasa** (ang. *flipped classroom*), to podejście do prowadzenia zajęć w szkole zainicjowane przez J. Bergmana i A. Samsa w 2012 roku [2]⁷. Polega na takiej organizacji uczenia się, w której edukacyjne aktywności uczących się poza regularnymi lekcjami, np. w domu, są wykorzystywane do lepszego przygotowania się do zajęć, w konsekwencji czas na zajęciach w szkole może być lepiej spożytkowany przez nauczyciela do zaspokojenia indywidualnych potrzeb uczniów. Taki tryb pracy uczniów poza szkołą może być uznany za zadanie domowe. Elementy odwróconego uczenia się można również odnaleźć w metodzie projektów.

Organizacyjnie, kształcenie w tym modelu powinno przebiegać w środowisku wirtualnym ze stałym dostępem uczniów do kanałów komunikacyjnych i zasobów (filmów i zadań wymienianych z nauczycielem i innymi uczniami), traktowanym jako spersonalizowane środowisko kształcenia zarządzane przez uczniów, wstępnie jednak zaprojektowane i wypełnione przez nauczyciela, by nauka poza lekcją znajdowała odniesienie do tego, co uczeń już zna i aby na tym bazował jego dalszy rozwój.

⁷ Podobne podejście reprezentuje Strategia Kształcenia Wypredzającego opisana w pracy [5].

11. Godzina kodowania, code.org

Godzina Kodowania (ang. *the Hour of Code*) to inicjatywa, zapoczątkowana w grudniu 2013 roku podczas Tygodnia Edukacji Informatycznej w Stanach Zjednoczonych. Inicjatorem był Hadi Partovi, który określił jej misję jako:

Każdy może nauczyć się informatyki (ang. *Anyone can learn Computer Science*).

Z latami ta inicjatywa przerodziła się ogólnościowy ruch, którego celem jest zachęcenie uczniów do nauki informatyki i zachęcenie szkół do regularnego oferowania uczniom zajęć z informatyki. Inicjatywa startowała z zestawami łamigłówek przewidzianymi na godzinę zajęć, a obecnie oferuje na stronie <https://code.org/> setki różnych aktywności w blisko 180 językach w tym w języku polskim, związanych z programowaniem blokowym. Uczniowie mogą je rozwiązywać przez cały rok.

Inicjatywa ta jest bardzo popularna w naszym kraju – w 2018 roku na code.org zarejestrowanych było ponad 650 tys. uczniów z Polski. Taką popularność zawdzięcza wielu przemyślanym rozwiązaniom: (1) bohaterami łamigłówek są postacie znane uczniom z ulubionych bajek, komiksów i gier; dodatkowo w code.org mogą wchodzić w interakcje ze swoimi bohaterami; (2) łamigłówki są w zestawach zadań o rosnącym stopniu trudności; najniższy poziom jest przeznaczony dla dzieci, które jeszcze nie potrafią czytać; (3) rozwiązanie łamigłówki polega na ułożeniu programu w języku blokowym (podobnym do Scratcha) do scenariusza przygotowanego w systemie; (4) uczniowie mogą wielokrotnie tworzyć, uruchamiać, debugować i ulepszać swoje programy; (5) mogą również przeglądać programy w JavaScript odpowiadające rozwiązaniom opartym na blokach. Uczniowie rozwiązując łamigłówki z code.org poznają kolejne konstrukcje algorytmiczne i programistyczne, stosują abstrakcję i dopasowywanie wzorców, następnie dekompozycję i wreszcie algorytmy rozwiązań łamigłówek. Dodatkowo, stosując zdarzenia, mogą programować interakcję, co jest dość zaawansowanym tematem informatycznym.

12. Informatyka z urządzeniami fizycznymi (ang. *physical computing*)

Pierwsze dwie dekady tego wieku to bujny rozwój **robotyki edukacyjnej**, która znakomicie współgra z konstrukcjonistycznym uczeniem się. Rozpowszechniły się także mikrokontrolery, a wśród nich układ (płytką) **BBC micro:bit**, zaprojektowany specjalnie dla celów edukacyjnych, zwany zresztą *brytyjskim komputerem szkolnym*. Stoi za nim idea, by to uczeń zbudował swój komputer, a nie siedział przed tym nudnym zestawem ekran + klawiatura (w luźnym przekładzie to słowa, jakie wypowiedział brytyjski minister edukacji Michael Gove w 2012 roku, gdy zapowiadał zerwanie w szkołach z przedmiotem technologia informacyjna). Tak zrodziła się **informatyka z urządzeniami fizycznymi** (ang. *physical computing*), której zajęcia, jeszcze mało popularne w Polsce,

polegają na jednoczesnym projektowaniu i budowaniu „komputera” dla określonego zadania oraz tworzeniu programu z bloków lub w języku Python, który będzie nim sterował. Takie urządzenie to „namacalny” artefakt uczenia się, często oryginalny i niepowtarzalny, osiągany w „poczuciu panowania nad fragmentem najnowocześniejszej techniki i w kontakcie z najgłębszymi ideami nauk przyrodniczych, matematyki i sztuki budowania intelektualnych modeli.” (Papert 1980, s. 25).

Informatyka z urządzeniami fizycznymi poszerza znaczenie pojęć informatycznych i obszar zastosowań informatyki. Dużą popularnością cieszy się inicjatywa *do your :bit* (zrób swój :bit), promująca wykorzystanie układu BBC micro:bit w systemach wbudowanych, związanych z ochroną środowiska i realizacją Globalnych Celów Zrównowagowanego Rozwoju⁸. W odniesieniu zaś do – wyzwala w nich kreatywność do tworzenia czegoś przydatnego w ich środowisku. Wraz z inteligentnymi obiektami sterowanymi sztuczną inteligencją, nowe treści stają się przystępne dla kształcenia informatycznego w szkołach na każdym poziomie [6].

Płytki BBC micro:bit, jako jeden podstawowy układ i jednocześnie środowisko programowania blokowego i tekstowego, może zostać wykorzystana w spiralnym rozwoju umiejętności uczniów w zakresie programowania w klasach IV-VIII.

Literatura

1. Being Fluent with INFORMATION TECHNOLOGY, National Academy Press, Washington, D.C., 1999.
2. Bergman J., Sams A. (2012), Flip your Classroom. Reach Every Student in Every Class Every Day, ISTE.
3. Bruner J.S (1964), Proces kształcenia, PWN [The Proces of Education, 1977].
4. Dewey J. (1988), Jak myślimy, PWN; Dewey J. (1963), Demokracja i wychowanie, KiW.
5. Dylak S. (2013), Architektura wiedzy w szkole, Difin, Warszawa.
6. <https://microbit.org>.
7. <https://www.csunplugged.org>.
8. Klus-Stańska D. (2018), Paradygmaty Dydaktyki. Myśleć teorią o praktyce, PWN.
9. Korczak J., Wojdyła J. (1981), System wspomaganego komputerem nauczania OSKAR, Prace Naukowe AE, Wrocław, nr 181/, 115-126.
10. Kupisiewicz Cz. (1966), Nauczanie programowane, PZWSz, Warszawa.
11. Papert S. (1996), Burze mózgów. Dzieci i komputery, PWN [oryginał, 1980].
12. Papert S. (1970), Teaching Children Thinking, WCCE 1970, Amsterdam.
13. Phillips D.C., Soltis J.F. (2003), Podstawy wiedzy o nauczaniu, GWP;

⁸ Na początku 2022 roku z płytki BBC micro:bit korzystało ponad 60 mln uczniów na całym świecie.

14. Skinner B.F. (1954), The science of learning and the art of teaching. *Harvard Educational Review* 24, 86.
15. Skinner B.F. (2003), The Technology of Teaching, B.F. Skinner Foundation.
16. Siemens G. (2005), Connectivism: A Learning Theory for the Digital Age, *International Journal of Instructional Technology and Distance Learning*. 2; 1–9.
17. Siemieniecki B. (2013), *Pedagogika kognitywistyczna. Studium teoretyczne*, Impuls.
18. Sysło M.M. (2023), Informatyka z komputerem w tle, W cyfrowej szkole, 42-52.
19. Sysło M.M. (2018), Jak myśleć komputacyjnie, *IwE XV*, 3-14.
20. Sysło M.M. (2024), Pascal as the Driving Force of Informatics Education in Schools in Poland, *Informatics in Education*, 4/23, 866–879.
21. Sysło M.M. (2023), Przyszłość Laboratorium, W cyfrowej szkole, 1/2023, 16-29
22. Walat A. (2007), O konstrukcjonizmie i ośmiu zasadach skutecznego uczenia się według Seymoura Paperta, *MERITUM* 4(7).
23. Wing, J.M. (2006), Computational Thinking. *Comm. ACM* 49(3), 33–35.