dr inż. Zbigniew Ledóchowski

Pomorska Liga Zadaniowa 2022 – informatyka
POWRÓT DO NORMALNOŚCI?

Szósta edycja Pomorskiej Ligi Zadaniowej po raz pierwszy od 3 lat została przeprowadzona w taki sposób, jak to miało miejsce przed pandemią. Jedynie jesienią 2021 były niewielkie znaki zapytania związane z I etapem (w związku z jesienną falą pandemii), ale ostatecznie zarówno ten, jak i kolejne dwa etapy były już właściwie niezagrożone tzn. zostały przeprowadzone tak, jak to miało miejsce do roku 2019. Tytułowy znak zapytania to swego rodzaju asekuracja, choć także wyrażona wątpliwość. Czy już możemy spokojnie ogłaszać trwały powrót do starych „normalnych" zasad organizowania konkursu? Od strony epidemiologicznej czas pokaże, natomiast w tym tekście spróbujemy poruszyć kilka zagadnień związanych z powracającą „normalnością" w sensie merytoryczno-organizacyjnym.

Niezależnie od tego, w jakiej formie przeprowadzany był konkurs w latach poprzednich, nie zmieniała się jego ogólna koncepcja programowa. Nie inaczej było w roku 2022, kiedy zadania były konstruowane wg tego przyjętego schematu. Oznacza on, przypominając rzecz pokrótce, że w etapie I (dla obu typów szkół) zachęcamy uczestników do aktywnego udziału w konkursie tworząc pytania w większości testowe, odnoszące się do wiedzy, którą powinni wynieść ze szkolnych zajęć informatyki. Uczniowie na ogół dobrze radzą sobie z tymi pytaniami, nawet jeśli wśród pytań znajdą się pojedyncze odwołujące się mniej do programów szkolnych, a bardziej do zainteresowań informatycznych uczniów rozwijanych indywidualnie. Etap I (realizowany poza pracowniami szkolnymi) uzupełniają nietrudne typowe zadania o charakterze algorytmicznym. Etap II to zadania o wiele bardziej rozbudowane, wymagające poświęcenia na ich rozwiązanie większej liczby czasu w warunkach domowych (etap ma charakter korespondencyjny). Poczesne miejsce zajmują problemy związane z algorytmami i programowaniem (o stopniu trudności adekwatnym do typu szkół), ale mamy też zadania wymagające przetwarzania danych przy pomocy oprogramowania użytkowego. Tradycyjnie dla szkół podstawowych pojawia się (niejako zachęcająco) zadanie wymagające szeroko rozumianej edycji dokumentów multimedialnych, często połączone ze sprawdzeniem wiedzy, której elementy uczniowie we wspomnianych dokumentach umieszczają. Największym wyzwaniem, o czym w tym miejscu pisano niejednokrotnie jest konstrukcja zadań finałowych. Staramy się zatem, aby i na tym etapie pojawiły się zadania wymagające sformułowania w postaci algorytmicznej (i zapisu w postaci programu) rozwiązania pewnych problemów. Problemy powinny być adekwatne swoim stopniem trudności dla etapu finałowego, ale też nie powinny być zbyt czasochłonne, bo 90 minut trwania finału to jednak dość poważne ograniczenie. Chodzi zatem o problemy wymagające raczej błyskotliwości, ale i dużego wyrobienia w świecie algorytmów. Oczekuje się na ogół rozwiązania wyłącznie w postaci algorytmu, nie programu. Oczywiście w etapie finałowym jest też miejsce na odpowiednio dla tej części „skrojone" problemy, związane z przetwarzaniem danych za pomocą oprogramowania użytkowego oraz związane z redakcją dokumentów o różnym charakterze (dla uczniów szkół podstawowych).

Teraz o kilku elementach szczegółowych, które w roku 2022 wiązały się z tą ogólną koncepcją. Po pierwsze już drugi rok z rzędu przyjęto dość ryzykowne założenie, że zadania etapu finałowego nie muszą być koniecznie przez zwycięzcę rozwiązane bezbłędnie oraz w całości. Niech zwycięży ten, kto poprawnie rozwiąże ich jak najwięcej. Obszerne uzasadnienie tego założenia zostało w tym miejscu przedstawione w ubiegłym roku. Tyle, że wtedy etap finałowy był przeprowadzony on-line. Czy tego typu podejście broni się w warunkach stacjonarnych? Oczywiście można mieć wątpliwości, czy szerokiej rzeszy uczniów nie zagwarantować (poprzez odpowiedni dobór treści zadań) możliwości rozwiązania wszystkich problemów. Coraz trudniej jednak dla zdolnych uczestników PLZ dobierać akurat w informatyce problemy zwięzłe (czas rozwiązania) i trudne, jak dla etapu finałowego. Wydaje się, że zarysowana koncepcja zadań „na styk", czyli tak dobranych, aby tylko najwybitniejsi mieli szansę rozwiązać je w 90 minut jednak się broni, także w warunkach konkursu organizowanego stacjonarnie – przynajmniej w informatyce. Inne wyjście to więcej czasu dla uczestników finału (chyba

mało realne ?), albo zmiana koncepcji w zakresie tego czego oczekujemy od rozwiązań (wspomnimy o tym dalej w odniesieniu do szkół podstawowych), ale to też nieco ryzykowny pomysł.

Kolejna sprawa to same treści zadań. W przypadku wspomnianego zadania związanego z redakcją dokumentów multimedialnych dla szkół podstawowych, jak wspomniano wyżej staramy się, aby umiejętności czysto techniczne były powiązane z wiedzą (w etapie II wręcz pozyskiwaną z różnych źródeł i niekoniecznie tylko informatyczną). I dlatego w tym roku budując dokument i prezentację w etapie II uczniowie mogli przy okazji poszerzyć wiedzę o krajach skandynawskich. Z kolei w finale wiedza była bardziej „hermetyczna", za to wymagała rozpoznania znanych (i czasami mniej znanych, jak pokazały rozwiązania) elementów z różnych dziedzin – szeroko rozumianej informatyki i nowych technologii.

Uczniowie szkół ponadpodstawowych, a przynajmniej uczestnicy PLZ z informatyki nie mają większych kłopotów z posługiwaniem się chociażby arkuszem kalkulacyjnym. Coraz mniejszy sens ma zatem układanie zadań powtarzających nieco schematy maturalne (nawet jeśli odwołujących się do bardziej subtelnych funkcji i narzędzi arkusza). Nie szukano zatem tym razem zadań opierających się na danych z pliku tekstowego, które trzeba poddać takim, czy innym operacjom (statystycznym, graficznym itp.), w których struktura rozwiązania była w pewnym sensie z góry określona przez postać danych, ale raczej zadań problemowych, w których to jak powinien wyglądać schemat obliczeniowy, czy też struktura arkusza zależy wyłącznie od rozwiązującego. Wydaje się, że w miarę to się udało. W etapie II pojawiło się zadanie, w którym mowa była o tym, aby skonstruować pomoc dla anonimowego wykładowcy, który przeprowadza test dla swoich studentów i w którym to teście są pytania zarówno jednokrotnego, jak i wielokrotnego wyboru. Skonstruowany arkusz miał automatyzować proces wprowadzania wyników egzaminu, sprawdzania poprawności testu i wystawiania ocen (przy dość specyficznych kryteriach oceny jakie stosował wykładowca), a nadto generować na użytek wykładowcy raport wyników w narzuconej formie. Poza tymi założeniami w treści zadania nie było żadnej sugestiijak arkusz ma wyglądać, co powinno się w nim w jakim miejscu znaleźć. To dało pożądany efekt, bo wśród rozwiązań poprawnych nie było praktycznie identycznych, co do formy samego arkusza. Podobnie było w finale, kiedy uczniowie mieli zaimplementować właśnie w arkuszu kalkulacyjnym znaną im zapewne z zajęć metodę połowienia szukania zer funkcji (bez używania makropoleceń!). Narzucenie arkusza (podobnie jak w omówionym zadaniu z etapu II) jako narzędzie nie jest może w zadaniach związanych z przetwarzaniem danych najlepszym pomysłem, bo wybór optymalnego narzędzia dla rozwiązywanego problemu jest jedną z umiejętności, której oczekujemy od uczestników PLZ, ale tym razem taki zabieg musiał mieć miejsce. W przypadku etapu II chodziło o przejrzystość wizualną rozwiązania z punktu widzenia jego przyszłego użytkownika (czyli fikcyjnego wykładowcy). W przypadku etapu finałowego nie chodziło zaś o powtórzenie implementacji znanego algorytmu w dowolnym języku programowania, bo wtedy problem nie byłby w ogóle godny etapu III jako problem odtwórczy. Cała trudność zadania polegała na zaimplementowaniu iteracji w arkuszu, co bez używania makropoleceń nie jest na ogół ze względu na naturę arkusza zadaniem banalnym.

W przypadku zadań algorytmicznych związanych z programowaniem poza jak zwykle inwencją własną oczekiwano od uczniów znajomości wielu gotowych rozwiązań, czyli pewnych algorytmów, które powinny były być im znane ze szkoły, a tu mogły być przydatne w rozwiązaniu konkretnych problemów. I tak w przypadku szkoły podstawowej w etapie II i III można było skorzystać z takich algorytmów jak zamiana liczby zapisanej w systemie dwójkowym na system dziesiętny, znajdowanie największej liczby w ciągu liczby, iteracyjne generowanie wartości ciągu wg określonego wzoru, proste algorytmy geometryczne związane z położeniem punktu w układzie współrzędnych, podział liczby naturalnej na cyfry. W odniesieniu do szkoły ponadpodstawowej przydać się mogły (niekiedy w szerszym kontekście) takie algorytmy jak: algorytm badania czy dwa słowa są swoimi anagramami, algorytm Rabina-Karpa wyszukiwania wzorca w tekście, podstawowe algorytmy geometryczne, algorytmy na ciągach liczbowych np. znajdowanie najdłuższego podciągu spójnego o zadanej własności. W praktyce z tym wykorzystaniem gotowych rozwiązań było różnie, podobnie jak z zadaniami odwołującymi się do zagadnień spoza podstawy programowej. W tej drugiej kwestii lepiej jednak wypadli uczniowie szkół ponadpodstawowych w zadaniu z zakresu modelowania i symulacji, niż uczniowie szkół podstawowych, którzy praktycznie w ogóle nie poradzili sobie z problemem rekurencyjnym. Za to generalnie należy pochwalić uczniów obu typów szkół za pewną biegłość w posługiwaniu się strukturami danych i biegłość, a nawet pomysłowość w ich doborze do treści problemu

(naturalnie z poprawką, że uczniowie szkół ponadpodstawowych posługują się strukturami danych lepiej, boi znają ich więcej).

Skoro mowa o zadaniach z programowania, na które niezmiennie stawiamy w PLZ (niezmieniające się uzasadnienie dla tego podejścia było już w tym miejscu formułowane) to powtarza się, a nawet coraz bardziej uwypukla pewien niezbyt budujący wniosek. Z zadaniami tymi coraz więcej kłopotu mają uczniowie szkół podstawowych. Już w ubiegłym roku wykonanie innych zadań niż te związane z algorytmami i programowaniem było podczas etapu powiatowego wystarczające do uzyskania wysokiego miejsca i awansu do finału, podobnie było w etapie wojewódzkim. Obecnie wspomniane zjawisko jeszcze bardziej się pogłębiło. W etapie powiatowym trzy zadania odnosiły się do implementacji algorytmów w postaci programów. Połowa uczestników tego etapu nie rozwiązała żadnego z nich! Maksymalną przewidzianą liczbę punktów uzyskano tylko w przypadku jednego z tych zadań (dwie osoby), a jeszcze inne zadania rozwiązały (i to co najwyżej na 50-60 % możliwych do zdobycia punktów) trzy osoby. W finale w przypadku zadania, w którym wymagano tylko algorytmu było lepiej, ale w przypadku dwóch zadań, gdzie oczekiwano także programu równie niedobrze. Oczywiście znajdzie się sporo obiektywnych, niejako tłumaczących uczniów przyczyn tego stanu rzeczy. Po pierwsze stopień trudności zadań (to jednak konkurs, a nie typowe problemy algorytmiczne rozwiązywane w szkole), po drugie działanie w określonych warunkach (np. ograniczony czas i związany z tym stres w finale). To jednak nie wyjaśnia wszystkiego. I widać, że z tą dziedziną tak wyeksponowaną w nowej podstawie programowej w odniesieniu do SP pewien problem się pojawia. Nie możemy generalizować, bo to wymagałoby szerszej ewaluacji podstawy programowej w skali kraju, ale widać, że uczniowie mają trudności z algorytmami i programowaniem, a kiedy – co jest przywilejem konkursów – pojawia się kwestia zastosowania nabytej wiedzy i umiejętności w problemach mniej typowych to owe trudności dotyczą nawet najlepszych. Oczywiście nie da się w tak krótkim tekście przeanalizować wszystkich aspektów tego zjawiska, warto też unikać wniosków pochopnych, zbyt daleko idących. Wydaje się, że wiele zależy przede wszystkim od nauczycieli i ich przygotowania (że jest z tym różnie nie jest wielką tajemnicą zwłaszcza w odniesieniu do SP, a wymagania programowe wzrosły), od metodyki nauczania właśnie algorytmów i programowania. Schowanie się za najprostszym wnioskiem, że to dziedzina za trudna dla ucznia i należy ograniczyć jej tak szybkie i wczesne wprowadzanie byłoby zbyt proste. Z uporem będziemy powtarzać, że ta podstawa taka musi być, bo taki jest wymóg cywilizacyjny. Można się zastanawiać nad pojedynczymi zagadnieniami, które może faktycznie przyczyniają się do przeładowania podstawy programowej nauczania informatyki w szkołach podstawowych, ale nie sposób kwestionować ogólnej idei. Myślenie komputacyjne i idące za nim programowanie już w najmłodszym wieku nie jest naszym wynalazkiem, dziś zwykła alfabetyzacja komputerowa, jak to mówiono kilka lat temu już nie wystarcza. A przecież nasze dzieci i młodzież nie są słabsze intelektualnie. Po sukcesach młodych ludzi w Międzynarodowej Olimpiadzie Informatycznej, czy studenckich mistrzostwach świata w programowaniu powstało powiedzenie, że „informatyka jest specjalnością młodych Polaków". Nawet jeśli dotyczy to pewnej elity naszych dzieci i młodzieży, to jednak wydaje się, że nie można tłumaczyć kłopotów z algorytmami i programowaniem tylko faktem, że trudne i za szybko wprowadzone. Chyba znaczniej więcej powodów leży po szeroko rozumianej stronie dydaktyki i uczących. A wracając do naszego konkursu to dostrzegamy zasygnalizowany problem i oczywiście pojawiają się dylematy z tym związane. Próbujemy jednak konsekwentnie trwać przy obranej drodze i jednak nie wycofywać się z dotychczasowej liczby zadań o tej tematyce. Częściowo z powodów ogólnych opisanych wyżej, a części dlatego, że może naiwnie, ale liczymy iż nastąpi tu jednak przełamanie i poziom uczniów w zakresie algorytmów i programowania będzie wzrastał (w końcu podstawy programowej nie można oceniać po ledwie kilku latach jej funkcjonowania). Poza tym jest grupa osób (dotyczy to także, a może bardziej uczniów szkół ponadpodstawowych), która niekoniecznie pozytywnie przywitałaby gwałtowny odwrót od tematyki związanej z algorytmami i programowaniem i co za tym idzie jednak obniżenie ogólnego poziomu konkursu. W przypadku uczniów szkół podstawowych pomysł jest nieco inny, nie wycofywać się z dotychczasowej liczby, ani nawet stopnia trudności zadań z algorytmiki i programowania, ale umieścić w zestawach choć kilka zadań, które miałyby coś w rodzaju już wykonanego początku, albo wskazówki co do istotnych etapów rozwiązania lub też wreszcie przygotowane pewne struktury z których uczniowie mogliby następnie „złożyć" całość. Trudno w tym miejscu opisywać szczegóły rodzącego się dopiero pomysłu na kształt części zadań z algorytmów i programowania, ale może będzie to miało jakiś bardziej zachęcający wpływ na uczestników konkursu i mniej osób będzie rezygnować z udziału w etapie II, po

pomyślnym zakończeniu etapu szkolnego (zjawisko bardzo charakterystyczne zwłaszcza dla szkół podstawowych), a Ci którzy wystartują dalej będą w większym niż dotąd stopniu próbowali rozwiązywać te jakże ważne, rozwijające myślenie zadania. Naturalnie nadal pozostałaby część zadań formułowanych w dotychczasowy sposób, które należałoby rozwiązać wychodząc samodzielnie od ich specyfikacji. W każdym razie na pewno nie rozważamy w interesie nie tyle poziomu konkursu, ale i samych uczniów żadnej rejterady od zadań związanych z algorytmami i programowaniem. Struktura konkursu musi bardziej jeszcze niż zajęcia szkolne odpowiadać trendom dydaktycznym. A przecież w kolejce, o czym głośno mówią eksperci, są kolejne wyzwania i nie od razu, ale już niedługo podstawa programowa będzie musiała przejść ewaluację i nie dlatego, żeby ją tylko odchudzać, ale dlatego że do drzwi puka jeszcze śmielej robotyka (jeszcze śmielej, bo już jest i jest lubiana przez uczniów, co tworzy pewną furtkę do kształtowania umiejętności programowania – choć ze zrozumiałych względów, akurat PLZ, poza teoretycznymi aspektami, zadań z robotyki wprowadzić nie może, są wszakże inne konkursy typowo robotyce dedykowane), czy sztuczna inteligencja.

Na zakończenie tego w pewnym sensie stałego wątku związanego z algorytmami i programowaniem jeszcze ciekawe spostrzeżenie. Dotyczy ono bardziej uczniów ze szkół ponadpodstawowych, ale i u niektórych uczniów szkół podstawowych (zwłaszcza w związku z etapem powiatowym) rzecz była widoczna. Otóż uczniowie dość dobrze (czasami to mało powiedziane)operują strukturami danych i zaawansowanymi wykraczającymi poza szkolne zajęcia konstrukcjami programistycznymi w różnych językach programowania. Nie zawsze idzie w tym w parze umiejętność sformułowania rozwiązania bardziej złożonych problemów w postaci algorytmicznej. Wychodzi zatem czasem tak, że mamy do czynienia z wysoką znajomością techniczną samych narzędzi, ale gorzej z ich wykorzystaniem. Trochę to koresponduje z przytoczonym w tym miejscu w ubiegłym roku pytaniem ucznia dotyczącym regulaminowego dopuszczenia jeszcze innych języków programowania, w tym takich specyficznych jak np. przywołany wówczas Haskell. Trudno wyciągać daleko idące wnioski z tych obserwacji i sytuacji, ale może się wydawać, że dla niektórych osób celem jest poznawanie języków samych w sobie i to z różnymi niuansami związanymi z ich konstrukcją. Często idzie wręcz o języki, których nauczyciel nie używa w szkole. Pytania jakiego języka programowania będziemy się uczyć zadawane nauczycielom w szkole nie są rzadkością, tak jak i nieskrywane rozczarowanie, że nie będzie to ten, który już sami uczniowie na własną rękę (i to nieźle) poznali. Nikt nie zamierza czynić zarzutu z tego rodzaju pasji informatycznych i dążenia do programowania w różnych językach. Problem w tym, że w metodyce nauczania programowania w szkole język ma być przede wszystkim narzędziem, a nie celem w samym sobie. Narzędziem do implementacji problemów algorytmicznych, ułatwiającym poznawanie różnych algorytmów i kształtowanie myślenia algorytmicznego, narzędziem realizacji ważnych z punktu widzenia informatyki szkolnej zastosowań. Pewnie, że przy okazji poznajemy różne języki programowania, doskonalimy umiejętność programowania, ale to jednak nie to samo co wyłącznie dogłębne (ze wspomnianymi niuansami) poznawanie samych języków programowania. Takowe ma miejsce na wyższych studiach informatycznych i tam generalnie jego miejsce. Wszystko zatem trzeba ustawić w odpowiednich proporcjach. Kształtujmy w szkole umiejętność programowania w tym stosując różne, najnowsze języki programowania (i bardzo dobrze), ale w służbie czegoś szerszego, czym jest myślenie algorytmiczne (czy nawet myśląc o kontekście zastosowań narzędzi informatycznych w innych przedmiotach myślenie komputacyjne). Być może te zachwiane proporcje (nie u wszystkich naturalnie) są jedną z przyczyn niepowodzeń z zadaniami algorytmicznymi podczas konkursu? Przy tej okazji wspomnijmy wszakże, że w bieżącej edycji można było zaobserwować bardziej elastyczne posługiwanie się nawet doskonale znanymi językami programowania adekwatnie do problemu. Mniej było sytuacji, w których autor rozwiązania „na siłę" używał ulubionego języka programowania, tam gdzie prosiło się bardziej optymalne narzędzie (system zarządzania bazami danych lub arkusz kalkulacyjny).

W poprzednich edycjach zwracał uwagę problem nie czytania treści zadań ze zrozumieniem (lub nieuważne czytanie), co zazwyczaj miewa swoje konsekwencje, W najlepszym razie kosztuje rozwiązującego utratę tak potrzebnego czasu, w najgorszym punktów. Są to (bo miały miejsce także w bieżącej edycji) różne sytuacje. Czasami dotyczą pominięcia ważnych aspektów polecenia, czasami nie dołączenie tego co miało być elementem odpowiedzi. Tym razem, żeby nie omawiać problemu tylko ogólnie posłużymy się przykładem z etapu wojewódzkiego dla szkół ponadpodstawowych, bo dotyczył kilku osób. Zacytujemy początkowy fragment zadania 3, podkreślając pogrubieniem kluczowy fragment:

Dane są dwie liczby zapisane w systemie dwójkowym: a i b. Wiemy, ***że każda składa się z n zer lub jedynek i że żadna z nich nie rozpoczyna się od zera.***

W dalszej części zadania, w uproszczeniu rzecz ujmując, chodziło o napisanie algorytmu, który stwierdzi, czy liczba *b* jest dokładnie o 1 mniejsza od liczby *a*. Ważne jest jednak cytowane zdanie, bo w nim pojawiły się dwa założenia, których zadaniem było skrócenie czasu pracy uczniom. Chodziło o to, aby nie badali, która liczba jest dłuższa (bo w założeniu miały być równe) i nie analizowali co by było, gdyby jedna z liczb, albo nawet obie miały jako wiodącą cyfrę zero. Tymczasem przynajmniej kilku rozwiązujących nieuważnie odczytało to zdanie (tym ciekawsze, że to pierwsze zdanie tego zadania) i rozważało również te warianty zadania, które założenie pozwalało pominąć. Często nie psuło to poprawności całego problemu, ale były i takie osoby, które skupiwszy się na sprawdzaniu tego, czego sprawdzać nie musiały, nie sprawdziły w dalszej części algorytmu warunków rzeczywiście mających wpływ na to, czy *b* jest o 1 mniejsze niż *a*. To już kosztowało utratę punktów, a można się założyć, że wszystkich, nawet tych, którzy punktów nie stracili kosztowało utratę przynajmniej kilku dodatkowych minut.30 sekund więcej uważnego czytania, zamiast kilku minut. Niby niewielka strata, ale przy 90 minutach trwania samego konkursu każda minuta jest cenna i szkoda ją tracić z takiego powodu, jak nieuważne czytanie treści zadania. Na koniec tego przykładu dodajmy, że choć w treści zadania wyraźnie było wskazane, że pytamy czy *b* jest o 1 mniejsze od *a*, a nie odwrotnie to znalazł się i taki uczestnik, który na wszelki wypadek sprawdzał (znowu kosztem dodatkowego czasu naturalnie), czy aby *a* nie może być mniejsze o 1 od *b*, choć o to celowo nikt nie pytał.

***

Na koniec tego krótkiego omówienia wybranych kwestii związanych z informatyczną częścią PLZ w edycji 2021/2022 chcemy poruszyć kwestię oceniania rozwiązań uczniowskich. Chodzi naturalnie o etapy powiatowy i wojewódzki, bo prace w etapie szkolnym, na podstawie dostarczanego do szkół klucza oceniania sprawdzają sami nauczyciele. Nie poruszyliśmy na ogół szczegółowo tej kwestii, bo jest to teoretycznie sprawa techniczna, „kuchnia" konkursu. Z drugiej strony zawsze ma prawo budzić emocje, bo decyduje o sukcesach i nagrodach. W przypadku informatyki, co ma związek z kwestiami omówionymi wyżej (zwłaszcza stopniem trudności zadań) rzadko kiedy wyniki są maksymalne. Trudno jednak zgodzić się z zasłyszaną opinią, że surowe ocenianie jest powodem tego, iż uczniowie czasami rezygnują z udziału w konkursie, w jego części informatycznej. Należy wyraźnie odróżnić uzyskanie niezbyt czasami wysokiej liczby punktów (zwłaszcza w zestawieniu z wyobrażeniem znających swoich podopiecznych nauczycieli o tym, ile ich zdaniem powinni osiągnąć), ale która to punktacja wynika po prostu z wyżej niż w szkole zawieszonej poprzeczki wymagań, od sytuacji, w której ta niższa niż oczekiwano punktacja wynikałaby rzeczywiście z surowego, a jeszcze gorzej niesprawiedliwego oceniania. Jesteśmy więcej niż przekonani, że mamy tu do czynienia przede wszystkim z oceną adekwatną do stopnia trudności zadań i to ten stopień trudności ma prawo być czynnikiem zniechęcającym niektórych potencjalnych uczestników konkursu. Ocenianie zaś nie powinno odgrywać tej roli, gdyż staramy się by jego zasady były czytelne, konsekwentnie stosowane, bez cienia wątpliwości co do ewentualnej niesprawiedliwości. A surowość? Na pewno nie jest większa niż powinna być w takich, konkursowych sytuacjach. Wiele też zależy od typów zadań, szczególnie w zadaniach z algorytmów i programowania ocenianie jest znacznie bardziej liberalne niż w innych konkursach oceniających napisane przez uczniów programy. Żeby sprawę bardziej rozwinąć przedstawimy nieco szerzej wspomniane wyżej ogólne zasady oceniania w odniesieniu do różnych typów zadań.

Jeśli, któreś z zadań są oceniane dość rygorystycznie, to dotyczy to zadań dla uczniów szkół podstawowych odnoszących się do redagowania dokumentów o różnej strukturze i postaci (dokumenty tekstowe, graficzne, prezentacje multimedialne, itp.). Zakłada się, że są to umiejętności, które uczniowie powinni mieć szczególnie dobrze ugruntowane, wielokrotnie przećwiczone i nawet jeśli w zadaniach konkursowych trafiają się pewne „subtelności", a zwłaszcza prośba o czynności dodatkowe (np. odniesienie do wiedzy teoretycznej lub informacji znalezionych w sieci internetowej), to powinno wymagać się precyzji w tworzeniu i formatowaniu tego typu dokumentów. Dlatego nie ma przysłowiowego machnięcia ręką, gdy rozdzielczość obrazu wykazuje duże odstępstwa od wskazanej w zadaniu, tabela jest niedokładnie sformatowana, a link prezentacji nie prowadzi tam gdzie powinien. Akurat w tych zadaniach na takie umiejętności techniczne stawiamy w równym stopniu, jak na inne elementy.

W przypadku zadań nieprogramistycznych, związanych najczęściej z przetwarzaniem danych dla obu typów szkół mamy do czynienia z wieloma niezależnymi poleceniami. Każde ma swój wynik, niekiedy w postaci wykresu. Warto podkreślić niezależność tych poleceń. Staramy się, aby nie układały się w tzw. wiązki i aby niemożność rozwiązania jednego z poleceń nie wpływała na skuteczność rozwiązania innych poleceń. Dążymy też do sytuacji, w której każda czynność w danym poleceniu miała swoje odzwierciedlenie w punktacji. Niekiedy klucz oceniania, a zwłaszcza ostateczna liczba punktów, jaką przyznajemy za całe zadanie utrudnia takie podejście, ale jeśli tylko można unikamy poleceń, w którym mamy 2-3 czynności do wykonania, a za to można otrzymać 1 punkt. Jeśli jeszcze takie polecenia pojawiają się to dotyczą czynności powtarzalnych (np. oblicz średnią dla pewnych danych, ale i przy okazji dla innych – w istocie chodzi o umiejętność liczenia średniej za pomocą odpowiedniej funkcji lub formuły arkusza kalkulacyjnego), albo dość elementarnych. W konsekwencji przy ocenie tych poleceń obowiązuje niestety zasada wszystko albo nic (muszą być wykonane wszystkie czynności, aby wspomniany wyżej punkt otrzymać). W zadaniach związanych z przetwarzaniem danych ważnym elementem jest oczywiście to, czy uzyskane przez rozwiązujących wyniki są zgodne z poprawnymi. Nigdy jednak nie podchodzimy do oceny wysiłków ucznia zero-jedynkowo. Doceniamy (w ramach klucza oceniania) element poprawnego rozwiązania – bywa tak, że banalna pomyłka (np. w adresie komórki użytej w poprawnej formule) sprawia, że wynik ucznia różni się od poprawnego. Przy podejściu zero-jedynkowym (liczy się tylko wynik) oznaczałoby to 0 punktów, w przypadku naszego konkursu w ogromnej liczbie takich przypadków tak nie jest. I właśnie dlatego, a nie tylko po to, aby ślepo naśladować pomysły z innych konkursów czy egzaminów prosimy, aby dla omawianej grupy zadań w charakterze rozwiązania przedstawiać zarówno plik tekstowy z samymi wynikami liczbowymi, jak i plik, w którym uzyskano rozwiązanie informatyczne. Ten pierwszy jest potrzebny do sprawnego porównania wyników ucznia z poprawnymi, a ten drugi, by przyjrzeć się uczniowskiemu rozumowaniu. Metodzie rozwiązania przyglądamy się wnikliwie nawet jeśli wyniki z pliku tekstowego są w pełni zgodne z poprawnymi. Chodzi zarówno o uniknięcie przypadkowości, ale i sytuacji w której poprawny wynik jest efektem nie do końca poprawnego rozwiązania, co w informatyce jest możliwe. Mamy tu również wyjaśnienie dla informacji podawanej zawsze do wiadomości ucznia, aby oddawać zawsze, i pliki z wynikami, i rozwiązaniem informatycznym. Przy czym za brak pliku z wynikami „kara" jest symboliczna – ostatecznie te wyniki można na ogół znaleźć w prezentowanym pliku z rozwiązaniem, absolutnie natomiast nie uwzględniamy samego pliku z wynikami, jeśli nie ma tego z rozwiązaniem informatycznym i wydaje się, że nie ma potrzeby wyjaśniać zasadności tego podejścia.

No i wreszcie zadania algorytmiczno-programistyczne. Te polegające na napisaniu samego algorytmu w zasadzie pojawiają się tylko w finale i generalnie towarzyszy im zasada: liberalizm dla przyjętej notacji, ale precyzja w ocenienie poprawności. Chodzi o to, aby notacja i zbyt duże uproszczenia w zapisie nie sprawiały, że algorytm jest niedokładny, czy niepełny. Jeśli chodzi o programy, to nie jest żadną nowością, że do oceny ich poprawności stosujemy podobnie, jak w wielu innych przypadkach się to robi (np. Olimpiada Informatyczna) wcześniej przygotowane testy. Szczupłość czasowa i kadrowa sprawia, że nie ma nigdy tylu zestawów testowych, ile stosowałaby np. wspomniana Olimpiada Informatyczna. Z drugiej strony nasze zadania nie są aż tak rozbudowane, nie ma tu też zbyt wielu przypadków specjalnych, z ukrytym „podtekstem". Staramy się zatem tak dobierać dane testowe, aby testy badały przede wszystkim poprawność rozwiązania (dla różnych danych), tzw. przypadki brzegowe, czy też związane ze specyfiką zadania (np. funkcjonowanie rozwiązania dla dużych liczb, jeśli taki zakres danych dopuszczała specyfikacja problemu). Nie testujemy natomiast absolutnie złożoności obliczeniowej, która w innych konkursach jest często ważnym elementem oceny. Doceniamy jej rolę i zawsze podziwiamy rozwiązania o optymalnej złożoności (w zgłaszanych propozycjach zadania dodatkowego widać, że bardzo wiele osób ze szkół ponadpodstawowych ma świadomość znaczenie tej własności algorytmów), ale nie punktujemy na razie wyżej rozwiązań tylko dlatego, że są bardziej optymalne od innych. Może kiedyś i o to warto się pokusić choć w jednym, wybranym problemie, ale na pewno nie w odniesieniu do szkół podstawowych. Warto jeszcze koniecznie wspomnieć o dwóch kwestiach związanych z zadaniami programistyczno-algorytmicznymi. Testy, którymi posługujemy się przy sprawdzaniu rozwiązań nie są łączone w żadne bloki testów, a w szczególności nie obowiązuje dość „twarda" zasada, że jeśli rozwiązanie nie przejdzie jednego testu w danym bloku nastawionym na sprawdzenie pewnej cechy rozwiązania, to za cały blok uzyskuje się 0 punktów. W przypadku PLZ efekty realizacji każdego testu rozpatruje się niezależnie od wyników dla innych testów, a ocena zależy oczywiście od tego, ile testów rozwiązanie przeszło pozytywnie, ale też

ważne jest jak istotną cechę z punktu widzenia danego zadania test brał pod uwagę – bywa, że pojedynczy test (negatywny) kosztuje tylko 1 punkt na 10 przewidzianych dla danego zadania, ale można stracić na nim i 3 punkty, jeśli testowana była szczególnie istotna cecha rozwiązania, którą badał odpowiedni zestaw danych testowych. Druga kwestia. Stosujemy nie tylko testy. Choć to zadanie dość karkołomne, to staramy się dokonywać również dość ogólnej analizy kodu, aby z tego wyciągnąć wnioski, co do przyjętego algorytmu lub co do tego, w którym miejscu i jakie błędy popełnione przez programującego sprawiły, że pewne testy nie wypadają pozytywnie. Stąd organizatorzy konkursów o bardziej rozbudowanych problemach algorytmiczno-programistycznych w ogóle się go nie podejmują, zdając się wyłącznie na testy, bo nie mówimy o problemach typowych (za to często rozbudowanych) i pomysły na algorytmy są bardzo różnorodne, często zawikłane i pojawia się niekiedy słynne pytanie „Co autor miał na myśli ?". Mimo tych przeciwskazań, póki co dla dobra ucznia, ciągle w możliwym zakresie, prowadzimy analizę kodów dostarczonych programów. I bywa, że kod rozwiązania uzyskuje 2 punkty (na 10) choć gdyby analizować tylko wyniki testów nie otrzymałby żadnego. Zresztą w instrukcji dla ucznia jest napisane, że nawet jeśli kod się nie kompiluje (niestety rzadko, ale otrzymujemy i takie rozwiązania), to będziemy czytać kod i na skromną (bo nie może być ona duża, gdy program się nie uruchamia) liczbę punktów rozwiązujący będzie mógł liczyć. Przy sprawdzaniu tylko za pomocą testów takie rozwiązanie otrzymałoby naturalnie z urzędu 0 punktów. Opisane podejście do sprawdzania zadań algorytmiczno-programistycznych traktujemy jako skromną misję wspierania uczniów, zwłaszcza ze szkół podstawowych, by jednak odważyli się choć zarysowywać rozwiązanie problemu, by nie bali się tej dziedziny. Wydaje się, że sprawdzanie tylko za pomocą testów działałoby bardzo odstraszająco, niezależnie od innych opisanych wcześniej kwestii, dotyczących algorytmów i programowania. Być może i w przypadku PLZ czynniki obiektywne sprawią, że poleganie tylko na testach w przypadku zadań algorytmiczno-programistycznych stanie się koniecznością, ale póki tak nie jest zawsze zachęcamy – pokaż choć część swojego rozwiązania dla tych zadań, a jeśli będzie tam zarys czegoś sensownego, to raczej nie zostaniesz bez punktów, bo nie polegamy wyłącznie na testach i sprawdzaniu „pod wynik".

Na zakończenie przyjętym zwyczajem wyróżnijmy jeszcze te osoby, które w przekroju wszystkich etapów części informatycznej PLZ szczególnie zachowały się w pamięci. W przypadku szkół podstawowych jest to na pewno: Dawid Szymonowicz (Szkoła Podstawowa im. Bohaterów Grudnia '70 w Łęgowie) zwycięzca etapu wojewódzkiego oraz bardzo równo prezentujący się w etapie powiatowym i wojewódzkim Hubert Rybak (Zespół Szkolno-Przedszkolny nr 3, Szkoła Podstawowa nr 47 Gdynia). W przypadku szkół ponadpodstawowych wymienimy przede wszystkim Bartłomieja Krawisza (V Liceum Ogólnokształcące im. Stefana Żeromskiego w Gdańsku), zdecydowanego lidera tegorocznej edycji (drugie miejsce w etapie powiatowym i pierwsze miejsce w etapie wojewódzkim). Wszyscy wymienieni, ale i jeszcze kilka osób z obu szczebli, na pewno wykazało się wysokimi umiejętnościami, zwłaszcza w odniesieniu do zadań najtrudniejszych.

Tegoroczna edycja PLZ w zakresie informatyki przeprowadzona po 3 latach wreszcie w typowych warunkach cieszyła się podobną popularnością jak poprzednie. Nadal warto zabiegać o większą liczbę uczestników ze szkół podstawowych, o czym była mowa wcześniej. Na szczeblu szkół ponadpodstawowych jak się wydaje konkurs ma się nieźle. Liczba uczestników jest dość stabilna, poszerzyła się geografia miast, z których pochodzą laureaci PLZ, z przyjemnością wśród uczestników znajdujemy też nazwiska osób startujących choćby w renomowanej Olimpiadzie Informatycznej. Wierzymy, że tak będzie nadal, a ze swej strony niezmiennie podejmiemy wszelkie starania, aby konkurs służył popularyzacji informatyki i pozwalał uczniom rozwijać swoje pasje w tym zakresie.

dr inż. Zbigniew Ledóchowski
Ekspert Pomorskiej Ligi Zadaniowej *Zdolni z Pomorza* w zakresie informatyki. Wykładowca akademicki, Akademia Pomorska w Słupsku, Instytut Nauk Ścisłych i Technicznych, Pracownia Informatyki.