**Model odpowiedzi i schemat punktowania informatyka maj 2019 Arkusz I**

**Egzaminatorze!**

* *Oceniaj prace zdających uczciwie i z zaangażowaniem.*
* ***Stosuj przyjęte zasady oceniania w sposób obiektywny****. Pamiętaj, że każda merytorycznie poprawna odpowiedź, spełniająca warunki określone w poleceniu, musi zostać pozytywnie oceniona, nawet jeżeli nie została przewidziana w przykładowych odpowiedziach w zasadach oceniania.*
* *Konsultuj niejednoznaczne rozwiązania zadań z innymi egzaminatorami lub przewodniczącym zespołu egzaminatorów. W przypadku niemożności osiągnięcia wspólnego stanowiska, rozstrzygajcie na korzyść zdającego.*
* *Przyznając punkty, nie kieruj się emocjami.*
* *Informuj przewodniczącego o wszystkich nieprawidłowościach zaistniałych w trakcie oceniania, w tym podejrzeń o niesamodzielność w pisaniu pracy.*

**Model odpowiedzi i schemat punktowania**

**Informatyka 2019, Arkusz1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr zadania | | Oczekiwana odpowiedź | Liczba punktów | |
| 1 | 1.1. | Za prawidłową odpowiedź **– 5 punktów**  **za rozwiązanie szybsze niż liniowe (np. wyszukiwanie binarne)**  w tym za:   * prawidłowy warunek pętli – 1 punkt * prawidłowe wyznaczenie podziału ciągu liczb – 1 punkt * prawidłowe wyznaczenie początku podciągu liczb – 1 punkt * prawidłowe wyznaczenie końca podciągu liczb – 1 punkt * prawidłowe wyznaczenie pierwszego elementu parzystego w *A*[] (lub jego indeksu)– 1 punkt   Przykładowe rozwiązanie (c++)  *p* ←1  *k* ← *n*  **dopóki** *p* <*k* **wykonuj**  *s* ← (p + k) div 2  **jeżeli** *( A*[*s*]mod2 *=* 1 *)*  *p* ← s + 1  **w przeciwnym przypadku**  *k* ← s  *w* ← *A*[*p*]  (python)  def szukaj\_bin(A):  lewy, prawy = 1, n  while lewy < prawy:  środkowy = (lewy + prawy) // 2  if A[środkowy] % 2 != 0:  lewy = środkowy + 1  else:  prawy = środkowy  return prawy  **za wyszukiwanie liniowe – 3 punkty**  w tym za:   * prawidłowy przebieg pętli – 1 punkt * sprawdzenie warunku (parzystości) – 1 punkt * prawidłowe wyznaczenie pierwszego elementu parzystego w *A*[] (lub jego indeksu)– 1 punkt   Przykładowe rozwiązanie:  *p* ←1  **dopóki***A*[*p*] *mod* 2 *=* 1**wykonuj**  *p ← p +* 1  *w ← A*[*p*]  **Każde inne poprawne rozwiązanie o złożoności lepszej niż liniowa –max 5 pkt, a o złożoności liniowej – max 3pkt.**  Przykład rozwiązania poprawnego o złożoności pierwiastek z n (za 5 punktów)  int pier(){  int i = 1;  while(i \* i < n) i++;  if(i \* i > n) i--;  return i;  }  int wyszukiwanie(){  int p = pier() - 1;  int i = p;  while(i < n)  {  if(T[i] % 2 == 0){  int j = i;  while(T[j] % 2 == 0) j--;  return j + 1;  }  if(i + p > n) i = n - 1;  i += p;  }  W tym przypadku  1 punkt za prawidłowe policzenie pierwiastka z n o złożoności szybszej niż liniowa  2 punkty za wyznaczenie bloku, w którym jest szukana liczba; w  tym 1 za sprawdzenie ostatniego bloku  2 punkty za prawidłowe wyznaczenie szukanej liczby lub pozycji szukanej liczby; w tym 1 punkt za poprawne przejście bloku , 1 za wyznaczenie szukanej wartości | 5 | 6 |
| 1.2. | Za poprawną złożoność – **1 punkt**  Odp. log(*n*) lub logarytmiczna  Dla wyszukiwania liniowego złożoność liniowa | 1 |
| 2 | 2.1 | Za prawidłową odpowiedź – **2 punkty**  W tym za:   * poprawne uzupełnienie drzewka wywołań funkcji – 1 punkt (wywołania funkcji *pisz*) * prawidłową kolejność wywołań – 1 punkt   Odp.: | 2 | 6 |
| 2.2 | Za prawidłową odpowiedź – **2 punkty**  w tym:  za każde poprawnie uzupełnione 2 pola tabeli 1 punkt **uwaga:** teksty wypisane przez funkcję mogą być zapisane  w jednym wierszu  Odp.:   |  |  |  | | --- | --- | --- | | wywołanie funkcji | tekst wypisany przez funkcję pisz() | liczba wywołań funkcji pisz() | | pisz("", 3, 2) | 000 001 010 011 100 101 110 111 | 15 (1+2+4+8) | | pisz("", 2, 3) | 00 01 02 10 11 12 20 21 22 | 13 (1+3+9) | | 2 |
| 2.3 | Za prawidłową odpowiedź – **2 punkty**  W przypadku liczby mniejszej o 1, lub gdy ostatnim elementem jest kn-1 zamiast kn – 1 punkt  Odp.: (kn+1-1)/(k-1) lub (1- kn+1)/(1-k) lub 1+k+k2+…+kn  może być też zapis z użyciem znaku sumy | 2 |
| 3 | 3.1 | FPPF | 1 | 3 |
| 3.2 | PFPP | 1 |
| 3.3 | PPFF | 1 |