Grupa młodsza. Pętle

Sponsorem dzisiejszych zajęć jest instrukcja `while`, literka `x` i liczba `3`.

Rozgrzewka bez komputera

I.1. Wykonaj następujący program dla $x = 3$, $x = 4$ i $x = 7$ (na tablicy).

```
while (x != 1)
{
   if (x % 2 == 0) {
     x = x / 2;
   } else {
     x = 3 * x + 1;
   }
   cout << x << endl;
 }
```

I.2. Wyjaśnij działanie następującego programu (`b` i `c` to zmienne typu `int`).

```
int a = 0;
while (true)
{
    if (b == 0) break;
    if (b % 2 == 1) {  a += c; }
    b /= 2;
    c *= 2;
}
cout << a << endl;
```

Wskazówka. Wykonaj program dla kilku wartości b i c.

Wskazówka. Pomyśl o rozkładzie dwójkowym liczby b.

I.3. Wyjaśnij, co robi następujący program (`n` to zmienna typu `int`).

```
int a = 1;
for (; n > 0; a *= n--);
cout << a << endl;
```

I.4. Wyjaśnij różnicę pomiędzy programami

```
while (n > 0) {
  cout << "Czesc!" << endl;
  n--;
}
```

```
do {
  cout << "Czesc!" << endl;
  n--;
} while (n > 0);
```

I.5. Wyjaśnij działanie programu

```
for (int i = 0; i <= 10; i++)
{
    if (i % 2 != 0)
      continue;

    cout << i << endl;
}
```

Zadania na zajęcia

I.6. Napisz program wyświetlający n razy napis „Witamy na pokładzie!", za pomocą

a) pętli `for`

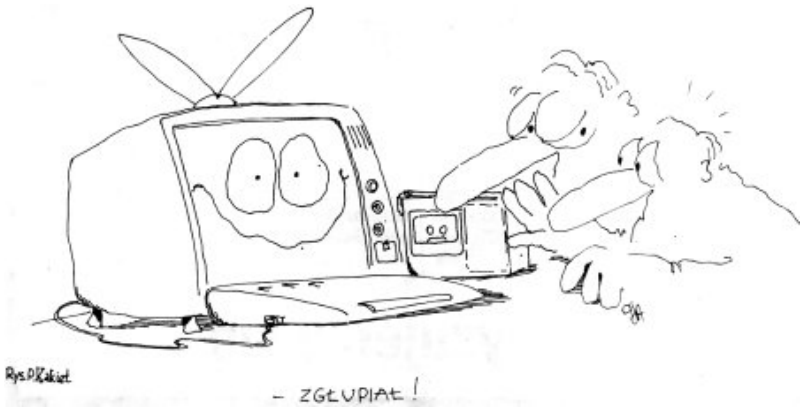
b) pętli `while`

c) pętli `do ...while`

I.7. Napisz program, który pyta o liczbę n a następnie o n liczb, których sumę wyświetla na ekranie.

I.8. Napisz program jak wyżej, ale obliczający średnią arytmetyczną podanych liczb.

I.9. Napisz program jak wyżej, ale obliczący najmniejszą i największą z podanych liczb.



Grupa starsza. Sito Erastotenesa

II.1. Na jakie pytanie odpowiada poniższy program? (`n` to liczba typu `int`)

```
int d = 2;
while (d * d <= n) {
        if (n % d == 0) break;
        d++;
}
if (d * d <= n) {
        cout << "nie" << endl;
} else {
        cout << "tak" << endl;
}
```

Ile maksymalnie obrotów pętli wykona powyższy algorytm?

II.2. Sito Erastotenesa to algorytm pozwalający szybko wyznaczyć liczby pierwsze z przedziału od 1 do n (szybko, to znaczy w czasie proporcjonalnym do $n \log\log n$. Iteracyjne wykorzystanie poprzedniej funkcji dałoby algorytm o złożoności proporcjonalnej do $n\sqrt{n}$. Dla $n = 10^6$ mamy $\log\log n < 1$ a $\sqrt{n} = 1000$).

Schemat algorytmu (sito Erastotenesa)

1. Utwórz tablicę `bool p[n+1]` i wypełń ją wartościami `true`.
2. `p[0] = false; p[1] = false;`
3. `d = 2;`
4. Dopóki `d * d <= n` wykonuj następujące operacje
   a) Dla `k = 2, 3,` ... dopóki `k * d <= n` podstawiaj `p[k * d] = false;`
   b) Znajdź następne d takie, że `d*d <= n` i `p[d] == true`
5. Wypisz wszystkie liczby `i`, dla których `p[i] == true`.

Algorytm ten wypisze wszystkie liczby pierwsze pomiędzy 2 a n. Wyjaśnij jego działanie i zaimplementuj go.

II.3. Napisz dwa programy, które obliczają ile jest liczb pierwszych od 2 do 8000000:
a) pierwszy wykorzystujący pierwszą metodę.
b) drugi wykorzystujący sito Erastotenesa.

Sprawdź czasy działania obydwu programów. Na moim komputerze jest to odpowiednio $5,21$ i $0,15$ sekundy. Oba algorytmy podają wynik 539777.

Prawo Murphy'ego

Jeżeli wydaje się, że wszystko działa dobrze, to z pewnością musiałeś coś przeoczyć.

Szukanie dzielników

```
int n; cin >> n;

int ile = 0;

for (int i = 2; i <= n; i++) {
  int d = 2;
  while (d * d <= i) {
    if (i % d == 0) break;
    d++;
  }
  if (d * d > i) { ile++;}
}

cout << ile << endl;
```

Sito Erastotenesa

```
int n;  cin >> n;

bool p[n+1];

for (int i = 0; i <= n; i++) { p[i] = true; }

p[0] = false; p[1] = false;

int d = 2;

while (d * d <= n) {
  for (int k = 2; d * k <= n; k++) {
    p[d * k] = false;
  }

  do {
    d++;
  } while (d * d <= n && p[d] == false);
}

int ile = 0;
for (int i = 0; i <= n; i++) {
  if (p[i]) {
    ile++;
  }
}
cout << ile << endl;
```