**Zadanie 1.**

Funkcja monitorująca działanie komputera uruchamia się regularnie co 30 sekund. Po każdym wykonaniu zwraca ona raport w postaci liczby szesnastkowej z przedziału od 00h do FFh. Liczba ta, po przedstawieniu w formacie binarnym, wykorzystuje każdy bit jako wskaźnik błędu. Wartość różna od 00h sygnalizuje wystąpienie błędów. Aby zidentyfikować przyczyny błędów, należy odnieść się do dokumentacji, gdzie kody błędów odpowiadają pozycjom znaczących bitów w odczytanej liczbie. Przykładowo, 01h oznacza błąd nr 1, a 03h wskazuje na błędy nr 1 i nr 2.

Stwórz program w dowolnym języku, który przeanalizuje zwróconą liczbę i wyświetli numery wykrytych błędów.

**Rozwiązanie 1.**

**Listing 1.** Przykładowe rozwiązanie w języku C:

#include <stdio.h>

int main(void)  
{

int Wynik = 0xff;  
 int i;

for (i = 1; i <= 8; i++)  
 {  
 if (Wynik & 1)  
 {  
 printf("Błąd: %d\n", i);  
 }

Wynik >>= 1;

}

return 0;  
}

**Zadanie 2.**

Należy stworzyć program kodujący tekst, który działa w ten sposób, że każdą kolejną literę przekształca na podstawie jej odległości od początku alfabetu oraz sumy przesunięć wynikających z poprzednich liter. Na przykład: tekst „aaaa” pozostanie „aaaa”, „bbbb” zostanie zakodowany jako „bcde”, a „kkkk” jako „kueo”.

Wskazówki:

1. Program powinien obsługiwać wyłącznie małe litery alfabetu łacińskiego.
2. Litera ‘a’ ma odległość 0, litera ‘b’ odległość 1, litera ‘c’ odległość 2, i tak dalej.
3. Po literze ‘z’ alfabet cyklicznie wraca do litery ‘a’.

**Rozwiązanie 2.**

**Listing 2**. Program napisany w C:

#include <stdio.h>  
#include <string.h>

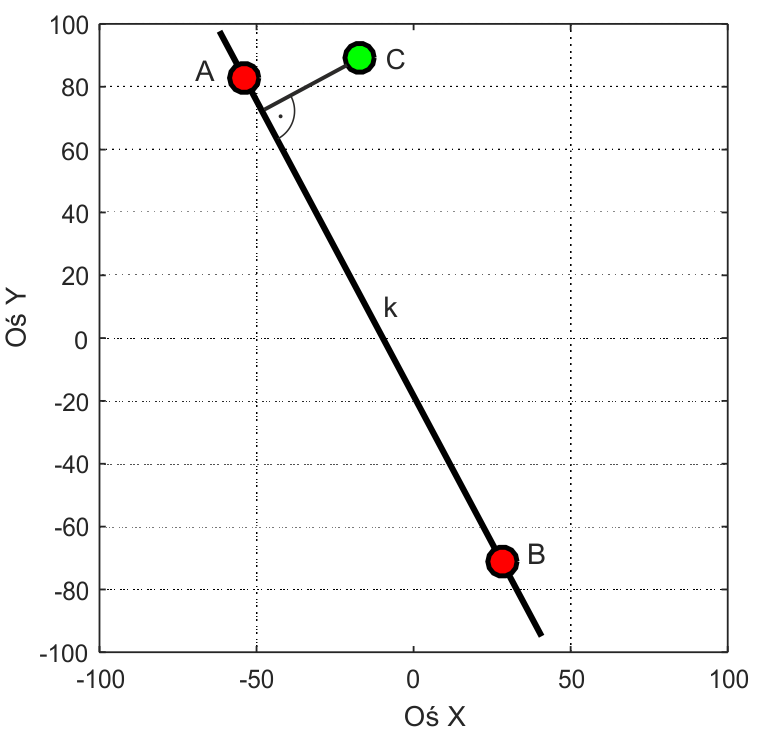
void zakoduj(char \*napis)  
{  
 int roznica = 0;  
 int i;  
 int dlugosc = strlen(napis);

for (i = 0; i < dlugosc; i++)  
 {  
 roznica = (roznica + (napis[i] - 'a')) % 26;  
 printf("%c ", (char)(roznica + 'a'));  
 }

printf("\n");  
}

**Zadanie 3.**

W dwuwymiarowym układzie współrzędnych dane są trzy punkty: A, B i C. Należy opracować program w wybranym języku programowania, który obliczy odległość punktu C od prostej k, wyznaczonej przez punkty A i B (zob. rysunek 1).



**Rys. 1.** Schemat do analizy położenia punktów A, B i C w dwuwymiarowym układzie współrzędnych

**Rozwiązanie 3.**

**Listing 3**. Przykładowy program napisany w C:

#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>

double odleglosc(double x1, double y1, double x2, double y2);

int main(void)  
{  
 double xA, yA, xB, yB, xC, yC, odl, u, xk, yk;

printf("Podaj kolejno wartość współrzędnych x i y dla punktu A: ");  
 scanf("%lf %lf", &xA, &yA);

printf("Podaj kolejno wartość współrzędnych x i y dla punktu B: ");  
 scanf("%lf %lf", &xB, &yB);

printf("Podaj kolejno wartość współrzędnych x i y dla punktu C: ");  
 scanf("%lf %lf", &xC, &yC);

/\* Obliczamy odległość między punktami A i B \*/  
 odl = odleglosc(xA, yA, xB, yB);

/\* Zgodnie z wzorem na rzut punktu C na prostą wyznaczoną przez A i B \*/  
 u = ( (xC - xA) \* (xB - xA) + (yC - yA) \* (yB - yA) ) / (odl \* odl);

/\* Wyznaczamy współrzędne rzutu punktu C (czyli xk, yk) \*/  
 xk = xA + u \* (xB - xA);  
 yk = yA + u \* (yB - yA);

/\* Obliczamy odległość od punktu C do prostej AB (czyli odległość C do punktu (xk, yk)) \*/  
 odl = odleglosc(xC, yC, xk, yk);

printf("Odległość punktu C do prostej przechodzącej przez punkty A i B wynosi: %.4f jednostek\n", odl);

system("pause");

return 0;  
}

double odleglosc(double x1, double y1, double x2, double y2)  
{  
 return sqrt( pow(x2 - x1, 2) + pow(y2 - y1, 2) );  
}

**Zadanie 4.**

Przedstaw schemat blokowy odpowiadający poniższemu fragmentowi kodu w języku C. Dodatkowo określ końcowe wartości zmiennych: wsp1, wsp2 oraz ilosc po zakończeniu działania algorytmu. Wyjaśnij również, jakie zadanie algebraiczne rozwiązuje przedstawiony kod.

**Listing 4.** Program w języku C realizujący pewne zadanie algebraiczne

#include <stdio.h>  
#include <math.h>

int main(void)  
{  
 int par1 = 2;  
 int par2 = 6;  
 int par3 = 1;  
 int ilosc = 0;

double wsp\_D = 0.0;  
 double wsp1 = 0.0, wsp2 = 0.0;

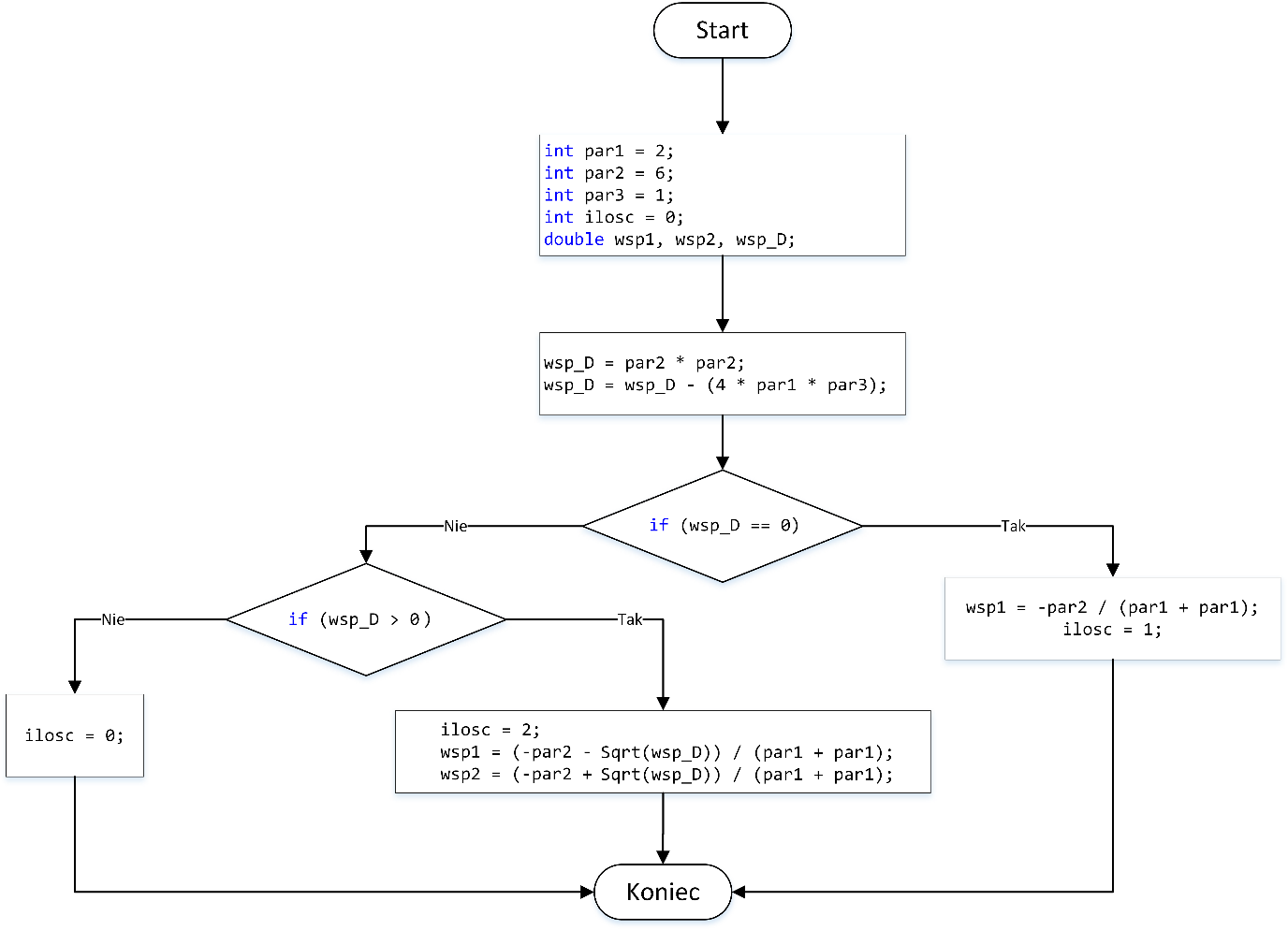
wsp\_D = (double)(par2) \* (double)(par2);  
 wsp\_D -= 4.0 \* (double)(par1) \* (double)(par3);

if (wsp\_D == 0.0) {  
 wsp1 = -par2 / (2.0 \* par1);  
 ilosc = 1;  
 }  
 else if (wsp\_D > 0.0) {  
 ilosc = 2;  
 wsp1 = (-par2 - sqrt(wsp\_D)) / (2.0 \* par1);  
 wsp2 = (-par2 + sqrt(wsp\_D)) / (2.0 \* par1);  
 }  
 else {  
 ilosc = 0;  
 }

return 0;  
}

**Rozwiązanie 4.**

Przedstawiony kod to algorytm wyliczania pierwiastków równania kwadratowego, operacje potęgowania zostały zastąpione mnożeniem, a wyrażenie „2a” dodawaniem „a + a”. Prawidłowy wynik zmiennych to wsp1 = -2,82, wsp2 = -0,18 i ilosc = 2. Przykładowy schemat blokowy przedstawia rys. 2.



**Rys. 2.** Schemat blokowy do rozwiązania równania kwadratowego

**Zadanie 5.**

Studenckie koło naukowe otrzymało od administratora sieci uczelnianej następującą podsieć: 192.168.0.0/17. Przewodniczący koła naukowego postanowił wykorzystać w pełni tą podsieć. Oto pytania na które należałoby odpowiedzieć przed przystąpieniem do administrowania tą siecią:

1. Jaki będzie adres sieci studenckiego koła naukowego?

2. Jaka powinna być maska sieciowa tej sieci?

3. Jaki będzie pierwszy dostępny adres dla hostów studenckich?

4. Jaki będzie ostatni dostępny adres dla hostów?

5. Ile hostów w tej nowej sieci przewodniczący koła studenckiego będzie mógł zaadresować?

6. Jaki będzie adres rozgłoszeniowy sieci (broadcast)?

**Rozwiązanie 5.**

1. 192.168.0.0;

2. 255.255.128.0;

3. 192.168.0.1;

4. 192.168.127.254;

5. 32766;

6. 192.168.127.255.