Zad. 1.

- Załóż katalog `programowanie` w którym będziesz pracować (`mkdir programowanie`) i przejdź do niego (`cd programowanie`)
- Za pomocą dowolnego edytora (np. Visual Studio) przepisz (nie kopiuj z pliku pdf) program jak poniżej:

```
#include<iostream>
using namepace std;

int main{

  cout << "Podaj liczbe całkowita= " << endl
  int liczba1
  cin >> liczba1;

  cout << Podaj liczbe rzeczywista=  << endl;
  double liczba2;
  cin >> liczba2

  wynik=liczba1/liczba2

  cout << "wynik= " < wynik << endl;

  return 0;
}
```

- Nadaj plikowi nazwę np. prog.cpp
- Skompiluj program za pomocą polecenia g++ (`g++ prog.cpp`)
- Do kompilacji dodaj flagę `-Wall`
- Popraw błędy i uruchom program
- Dodatkowo: sprawdź sam etap preprocesora i kompilacji (patrz - materiał wykładowy). Opcje polecenia `g++` można znaleźć pisząc `man g++`. Jakie pliki powstają we wszystkich trzech przypadkach ?

Zad. 2. Napisz program który:

1. wypisze na ekran:
   Podaj dwie liczby całkowite

2. wczyta te dwie liczby;

3. obliczy ich iloraz, resztę z dzielenia pierwszej liczby przez drugą, średnią arytmetyczną oraz geometryczną;

4. przed próbą dzielenia liczb przez siebie i liczeniem pierwiastka sprawdza poprawność wprowadzonych danych i w przypadku dzielenia przez zero lub pierwiastka z liczby ujemnej przerywa działanie programu (`return 1;`);

5. wypisze na ekran: `a/b =` dd
   `reszta z dzielenia a/b =` rr
   `Średnia arytmetyczna a i b =` xx
   `Średnia geometryczna a i b =` yy

Napisy `dd`, `rr`, `xx` i `yy` muszą być zastąpione wynikami obliczeń.
Do policzenia średniej geometrycznej użyj funkcji `sqrt` z biblioteki `cmath` (konieczne jest dodanie na początku pliku linii `#include<cmath>`).
Operator reszty z dzielenia to %
Przykładowo: 15%4 to reszta z dzielenia 15 przez 4.

Zad. 3. Napisz program znajdujący największą liczbę całkowitą spośród trzech podanych z klawiatury (i wypisujący na ekran jej wartość). Uwaga: spróbuj napisać jak najkrótszy algorytm do znajdowania największej liczby spośród podanych trzech (użyj tylko dwóch instrukcji warunkowych `if` bez `else`, bez && i bez ||).

Zad. 4. Napisz program szukający pierwiastków równania kwadratowego. Program powinien:

1. wypisywać na ekran komunikat:
   `Podaj współczynniki równania kwadratowego`
   a następnie wypisywać:
   `Podaj a`
   i wczytywać z klawiatury `a`.
   Analogicznie dla `b` i `c`.

2. Powinna być tworzona nowa zmienna `delta` i pod nią podstawiona wartość wyliczonej Δ równania kwadratowego. **Wypisz na ekran wartość obliczonej delty**.

3. Następnie sprawdzane powinny być trzy warunki : ($\Delta < 0$, $\Delta = 0$ i $\Delta > 0$) i w zależności od wartości Δ program powinien wypisywać na ekran wartości pierwiastków z komentarzem: `Rownanie nie ma rzeczywistych pierwiastkow`, `Rownanie ma jeden pierwiastek = ...` lub `Rownanie ma dwa pierwiastki: x1 = ...i x2 = ....`

Ważne – sprawdź czy program prawidłowo działa i znajduje pojedynczy pierwiastek dla a=1, b=-0.2 i c=0.01

Zad. 5. Napisz program, który wczytuje ze standardowego wejscia rok, a następnie wypisuje na standardowe wyjście czy ten rok jest przestępny czy nie. Na początku powinno być sprawdzane, czy nie podaliśmy liczby mniejszej lub równej zeru. W przypadku podania błędnych danych program powinien przerywać działanie z odpowiednim komunikatem (użyj polecenia `return`). Rok przestępny spełnia jeden z warunków:

- jest podzielny przez 4 z wyjatkiem lat podzielnych przez 100
- jest podzielny przez 400

Zad. 6. Wypisz na dwa sposoby na ekran liczby od 0 do n. n powinno być wczytywane z klawiatury. Skorzystaj z pętli `for` i `while` lub `do-while`. Obie pętle powinny znajdować się w jednym programie.

Zad. 7. Napisz program, który będzie wczytywał dwie liczby całkowite będące dolną i górną granicą przedziału liczb całkowitych i będzie zliczał i wypisywał na ekran ile liczb parzystych znajduje się pomiędzy wczytanymi liczbami. Po wczytaniu granic program powinien sprawdzać czy dolna granica jest mniejsza od górnej, a jeśli nie to powinien wypisywać komunikat, że jest problem i przerywać działanie za pomocą polecenia:
`return 1;`

Zad. 8. Napisz program sumujący kwadraty kolejnych n liczb naturalnych i wypisujący wynik na ekran. n podawane jest z klawiatury.

Zad. 9. Napisz program do liczenia n! Porównaj dla jak dużych liczb wynik ma sens w przypadku operowania na liczbach całkowitych i zmiennoprzecinkowych.

Zad. 10. Użyj **jednej** pętli `while` do wypisania na ekran obok siebie kolumn dwóch liczb: pierwszej z liczbami od 0 do n i drugiej z liczbami od n do 0. n powinno być wczytywane z klawiatury. Spróbuj stworzyć jak najkrótszy algorytm.

Zad. 11. Napisz program, który będzie prosił o podanie ze standardowego strumienia wejściowego dwóch liczb rzeczywistych i obliczał ich iloraz. Jeżeli

liczba przez którą dzielimy jest równa zeru, powinien prosić o podanie innej pary liczb i tak do skutku. Użyj nieskończonej pętli `while` i instrukcji `continue` i `break`.

Zad. 12. (`Dla zaawansowanych:`)
Napisz program, który będzie znajdował największy wspólny dzielnik dwóch liczb całkowitych. Znajdź w internecie na czym polega algorytm Euklidesa i go zastosuj.

Zad. 13. (`Dla zaawansowanych:`)
Napisz program sumujący wyrazy szeregu $1/10^i$, dla i od 0 do 20. Wypisuj chwilowe wartości sumy, po dodaniu każdego wyrazu. Po każdym wypisaniu na ekran dodaj przejście do nowej linii. W tym samym programie zrób dwie pętle różniące się tylko tym, że raz zmienna przechowywująca sumę będzie typu float, a raz typu double. **Porównaj liczbę cyfr znaczących w obu wypadkach.** Przy wypisywaniu wyników na ekran zwiększ liczbę wypisywanych cyfr do 20.
Liczbę wypisywanych cyfr można zwiększyć w następujący sposób:

```
#include<iostream>
#include<iomanip>
using namespace std;
int main(){
  ...
  cout << setprecision(20) << wynik << endl;
  ...
  return 0;
}
```

Zad. 14. (`Dla zaawansowanych:`)
Napisz program, który rysuje na ekranie kwadrat (prostokąt) o boku n (podawane z klawiatury), z zaznaczonymi obiema przekątnymi. Boki kwadratu i obie przekątne rysowane są dowolnymi znakami wczytywanymi z klawiatury. Polecenie wypisujące na ekran znak powinno pojawić się tylko raz.
Wskazówki:
Docelowo użyj dwóch struktur pętlowych.
Program powinien być pisany etapami:

- jeden wiersz
- pełny kwadrat
- sama ramka

- ramka z jedną przekatną
- ramka z dwiema przekątnymi

Zad. 15. (`Dla zaawansowanych:`)
Napisz program rysujący tabliczkę mnożenia, np. dla n=5 (n podawane z klawiatury):

```
1    2    3    4    5
2    4    6    8   10
3    6    9   12   15
4    8   12   16   20
5   10   15   20   25
```

Wskazówki Przy wypisywaniu wyników na ekran ustal stałą szerokość pola, i cyfry przesuń do prawej strony np.: `cout << right << setw(4) << n;`

Powtórzenie do kartkówki

1. Napisz program liczący sumę szeregu

$$\frac{1}{1\cdot 2}+\frac{1}{2\cdot 3}+\frac{1}{3\cdot 4}+\ldots+\frac{1}{n(n+1)}$$

   Zrób dwie wersje tego programu: z pętlą `for`, gdzie liczba dodawanych wyrazów szeregu jest wczytywana z klawiatury i z petlą `while` z warunkiem, że dodajemy kolejne elementy sumy, dopóki są większe od 10^{-5} (1e-5).

Zad. 16. Napisz dwie funkcje: jedną liczącą pole kwadratu o boku `a` i drugą liczacą pole koła o promieniu `r`. Funkcje nie powinny niczego wypisywać na ekran.
Napisz program, który będzie wypisywał na ekran pytanie : `Pole kwadratu czy kola ? (wpisz kwadrat lub kolo)`. Jeżeli użytkownik wpisał `kolo` to program powinien wypisywać pytanie o długość promienia, wczytywać promień i liczyć pole (korzystając ze stworzonej funkcji) i wypisywać pole koła. Jeżeli użytkownik wpisał `kwadrat` to program powinien wypisywać pytanie o długość boku, liczyć pole (korzystajac z funkcji) i wypisywać pole kwadratu. Do wczytywania i przechowywania napisów (nazw figur) użyj obiektu typu `string`. W przypadku podania błędnych danych program powinien przerywać działanie.

Zad. 17. Napisz funkcję `podzielna`:

```
bool podzielna(int liczba1, int liczba2)
```

sprawdzajacą czy liczba podana jako pierwszy argument funkcji jest podzielna przez liczbę podaną jako drugi argument funkcji i zwracająca typ logiczny (`bool`). Typ `bool` przyjmuje tylko dwie wartości: `true` lub `false`. Wewnątrz funkcji powinno być sprawdzenie czy `liczba2` jest równa zero. Jeśli jest, to uzyj funkcji `exit` z biblioteki `cstdlib` do przerwania programu (`exit(EXIT_FAILURE);`).
Napisz program, który będzie wczytywał liczbę której podzielność chcemy sprawdzić i liczbę która będzie dzielnikiem i przy użyciu stworzonej funkcji wypisywał na standardowe wyjście czy pierwsza z podanych liczb jest podzielna przez drugą. Funkcja nie może niczego wypisywać na ekran.

Zad. 18. Napisz funkcję, która będzie obliczać sumę wszystkich dzielników podanej liczby całkowitej. Napisz program, który używa tej funkcji dla liczb wczytanych ze standardowego strumienia wejściowego i wypisuje sumę jej dzielników na standardowy strumień wyjściowy.
Funkcja nie może niczego wypisywać na ekran.

Zad. 19. Napisz funkcję, która będzie sprawdzać, czy podana liczba jest liczbą doskonałą (suma jej dzielników właściwych jest równa danej liczbie). Np. 6 = 1+2+3
Przy użyciu tej funkcji znajdź i wypisz 3 pierwsze liczby doskonałe.

Zad. 20. Napisz dwie funkcje: `losujint`, która zwraca liczby losowe **(całkowite)** z przedziału [a,b] i `losujdouble`, która zwraca liczby losowe **(rzeczywiste)** z przedziału [a,b] Napisz program, który przy użyciu tych funkcji wypisze na ekran 15 liczb losowych z przedziału [0,10] i 15 liczb rzeczywistych z przedziału [0,1]. Zmodyfikuj program tak, aby wyniki były za każdym razem inne. Skorzystaj z funkcji `rand` i `srand` z biblioteki `cstdlib`.

Zad. 21. Napisz dwie funkcje **o tej samej nazwie**, które zamieniają wartości dwóch zmiennych podanych jako ich argumenty. Jedna funkcja ma zamieniać ze sobą dwie liczby rzeczywiste, a druga dwa znaki (`char`). Użyj obu funkcji w tym samym programie. Wypisz liczby i znaki przed i po zamianie.

Zad. 22. Napisz program, który za pomocą funkcji `getline` wczytuje ze standardowego strumienia wejściowego wyraz zapisany w całości małymi literami i sprawdza czy napis jest palindromem (wyraz brzmi tak samo czytany od lewej do prawej jak od prawej do lewej). Sprawdzanie czy dany napis jest palindromem powinno odbywać się w funkcji: `czy_palindrom`.
Wersja zaawansowana: program powinien sprawdzać czy całe zdanie napisane w całości małymi literami jest palindromem (spacje pomijamy). Przykład: `kobyla ma maly bok` jest palindromem.

Wskaźniki i tablice – zadanie na rozgrzewkę: