In dit document vind je oefenopgaven waarmee je programmeren kunt oefenen. De opgaven worden steeds iets moeilijker. Ga voor je zelf altijd na of je het probleem snapt. Om een goed programma te schrijven wat een lastiger probleem oplost, moet je het probleem in kleine stapjes oplossen. Bij de eerste opgaven geven we een aantal instructies welke stappen je het beste kunt nemen. Wanneer je verder komt, is het de bedoeling dat je zelf deze stappen kunt formuleren.

Houd een logboek bij waarin je de code en resultaten van elke opdracht kopieert.

Inhoud

Week 1

1. Leeftijd in maanden: Maak een programma wat iemands leeftijd in maanden berekent. Doe dit als volgt
    a. Declareer een int leeftijd
    b. Print een tekst op de console waarin om iemands leeftijd gevraagd wordt
    c. Lees de leeftijd uit met een scanf en sla deze op in leeftijd.
    d. Print een tekst waarin de leeftijd in maanden geprint wordt.
2. Gemiddelde: Maak een programma wat het gemiddelde van 3 integers kan berekenen. Doe dit als volgt:
    a. Declareer 3 integers, a, b en c.
    b. Print een tekst op de console waarin je vraagt om 3 integers in te voeren
    c. Lees elke integer uit met een scanf, of lees alle 3 de integers uit met 1 scanf
    d. Bereken het gemiddelde
    e. Print het gemiddelde op de console.
3. Max: Maak een programma wat van 2 integers die de gebruiker geeft de grootste waarde op de console print.
    a. Declareer integers, doe een printf en scanf vergelijkbaar als in opdracht 1 en 2
    b. Gebruik een if statement waarin je nagaat wat de grootste variabele is
    c. Print de juiste variabele op de console
4. Vul het programma van opdracht 3 aan zodat er een 3^e integer meegenomen wordt. De grootste integer van 3 integers wordt geprint. Gebruik hiervoor een if else statement.
5. Gelijkwaardigheid: Schrijf een programma waarin je kunt nagaan of twee integers gelijk aan elkaar zijn. Wanneer de integers gelijk aan elkaar zijn print je gelijk op de console, anders print je ongelijk.
6. Een float geeft een getal weer wat ook decimale waarden bevat. Wanneer we een float willen printen op de console kunnen we aangeven met welke precisie (significantie) we dat doen. Schrijf een programma waarin de gebruiker een float geeft die met 4 decimalen geprint wordt.
7. Schrijf een programma wat de omtrek en oppervlakte van een cirkel op basis van de radius berekent. Geef de radius van de cirkel mee met een scanf. Print deze waarden op de console.
8. BTW: Vraag aan de gebruiker om een prijs excl. Bereken de prijs incl. btw (21%)
9. Snelheidsconversie: Schrijf een programma waarin je de gebruiker vraagt om een snelheid in km/h en die omzet naar een snelheid in m/s. Gebruik hiervoor floats/doubles om je variabelen in op te slaan.
10. Wet van Ohm: Schrijf een programma wat op basis van een spanning en een weerstandswaarde de stroomsterkte kan berekenen.
11. Deling: Schrijf een programma wat zowel de grootste gemene deler en de rest van een deling kan berekenen. Voorbeeld 7/3 → Grootste gemene deler = 2, rest is 1. Voorbeeld 17/5 → Grootste gemene deler = 3, rest is 2.
12. Gegeven is een hoeveelheid seconden. Bereken hoeveel uur, minuten en (resterende) seconde dit zijn en print dit op de console.
13. Kwadrateren: Schrijf een programma waarin je de gebruiker een integer vraagt. Het kwadraat van deze integer wordt op de console geprint. Voorbeeld: 2 geeft 4.
14. 3^e machtsheffen. Maak een kopie van je programma bij 13 en voeg nu de mogelijkheid toe om 3^e machtsheffen. De gebruiker moet kunnen aangeven of hij wil kwadrateren, of wil 3^e machtsheffen.

Week 2

1. Wanneer een integer een te grote waarde krijgt, krijgen we een overflow. Dit betekent dat de integerwaarde in eens negatief wordt. Schrijf een programma wat twee positieve integers met elkaar vermenigvuldigt. Wanneer de uitkomst kleiner dan 214783647 is moet de uitkomst op de console geprint worden. Wanneer deze groter is (overflow) moet overflow op de console geprint worden.
2. Onderaan blz. 95 van je boek staat een stukje code waarin we de grootte van verschillende variabelen printen. Neem het stukje code over en voer het uit. Neem het resultaat over in je logboek. Tip: het kan nodig zijn om de sizeof eerst naar een int te casten. Je doet dit door (int) ervoor te zetten op elke regel. Bijvoorbeeld: (int) sizeof(char)
3. Maak een programma wat de cijfers 1 t/m 100 op de console kan printen. Gebruik hiervoor een for loop en print elk getal op een aparte regel.
4. Maak een programma wat de index van een char kan printen. Vraag de gebruiker om een char te geven. We printen dan niet het karakter zelf, maar de index (int) van dit charakter.
5. Maak een programma waarin je de som van een reeks cijfers van 1 t/m n kan berekenen. Vraag hiervoor n aan de gebruiker. Voorbeeld:
   N = 3 → 1 + 2 + 3 = 6
   N = 6 → 1 + 2 + 3 + 4 + 5 + 6 = 21
6. Schrijf een programma wat een integer optelt vanaf 0 totdat er een overflow plaatsvindt. Je hoeft het getal niet te printen. Gebruik hiervoor een while loop.
7. Maak een programma wat de gebruiker om 2 integers a en b vraagt. Het programma moet waar/niet waar printen op de console als a deelbaar is door b. Voorbeeld: 6 is deelbaar door 2, maar niet deelbaar door 4.
8. Vraag een letter aan de gebruiker. Print of de letter een klinker, danwel medeklinker is.
9. Gegeven is de prijstabel hieronder. Schrijf een programma wat de totale elektriciteitskosten kan berekenen voor 1 jaar. Vraag de gebruiker om zijn dal en piek verbruik.

Elektriciteit dal	€0,40 /kWh
Elektriciteit piek	€0,46 /kWh
Leveringstarief	€80 / y

Tip: Gebruik voor de volgende opdrachten de wiskundige functies (boek blz. 99). Vergeet niet math.h te includen.

10. Wortel voor 0-n: We willen de wortels van alle getallen tussen 0 en n hebben en deze op de console printen. Vraag de gebruiker om een positief getal n en print alle wortels tussen 0 en n.
11. Pas het programma bij 10 aan zodat de wortel van een getal gecast wordt naar een integer. Print de integer op de console
12. Tot de macht: Vraag de gebruiker voor twee integers, x en n. Print op de console de uitkomst van de volgende som: x^n.
13. Schrijf een programma wat kan bepalen of twee getallen op het laatste cijfer gelijk zijn. Voorbeeld: 17 en 227 hebben het laatste cijfer gelijk. 223 en 337 niet.
14. ABC-formule. Programmeer de ABC-formule $x_{1,2} = -\frac{b \pm \sqrt{b^2 - 4 \cdot a \cdot c}}{2a}$. De gebruiker geeft a b en c los van elkaar en de twee oplossingen van x worden als double geprint.

Tip: wanneer de discriminant ($b^2 - 4 \cdot a \cdot c$) kleiner is dan 0 krijg je een imaginaire oplossing. Vang deze situatie af met een if statement en print in dat geval de imaginaire oplossing.

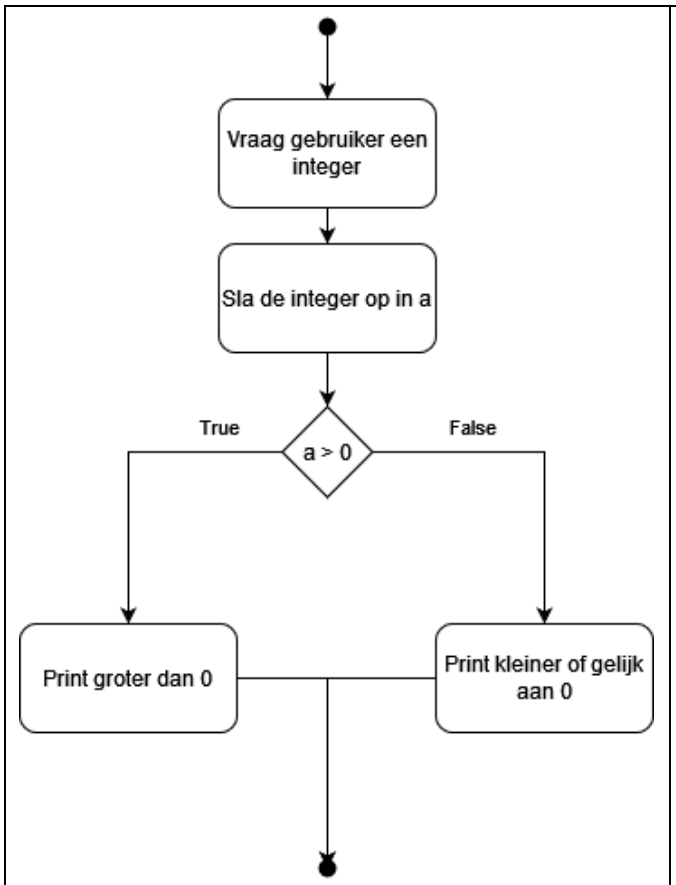
15. Schrijf een programma waarin je het volgende patroon kan printen (vraag n aan de gebruiker):

```
1
12
123
1234
.
.
.
1234…n
```

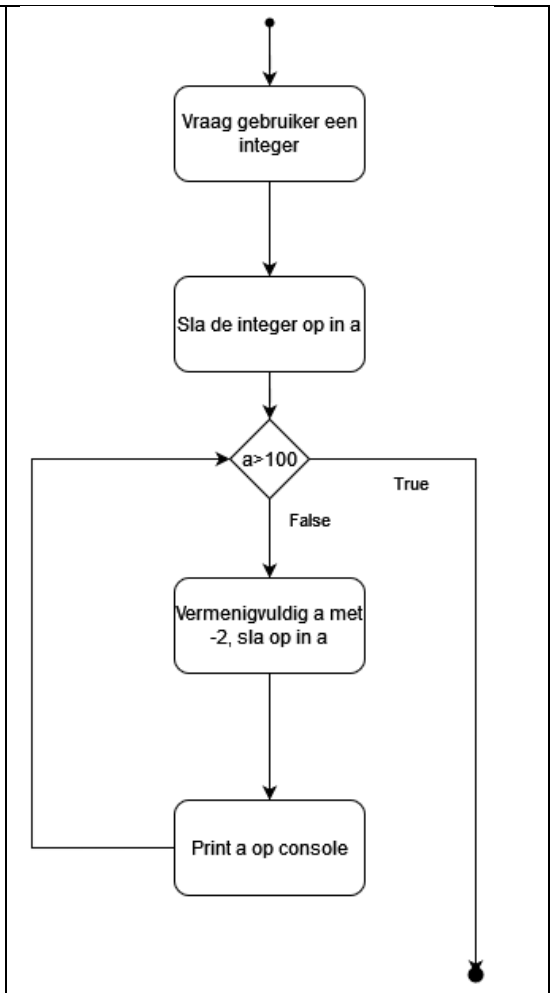
16. Schrijf een programma wat van een binaire reeks een decimaal getal kan maken.

Week 3

1. Programmeer Flowchart A
2. Programmeer Flowchart B



Flowchart A



Flowchart B

3. Gegeven is onderstaande code. Maak een flowchart waarin je de werking van de code beschrijft. Tip: Gebruik een editor zoals draw.io.

```c
#include <stdio.h>
int main() {
	int a;

	scanf_s("%d", &a);

	for (int i = 0; i < a; i++) {

		printf("Het kwadraat van %d is %d\n", i, i * i);

	}


}
```

4. Gegeven is onderstaande code. Maak een flowchart waarin je de werking van de code beschrijft.

```c
#include <stdio.h>
int main() {
	int a = -1;
	int b = -1;

	printf("Geef de lengte van zijde a en b van een rechthoek.\n");

	while (a < 1 || b < 1) {
		scanf_s("%d %d", &a, &b);

		if (a < 1 || b < 1) {
			printf("Een van beide zijden heeft een lengte van 0 of kleiner. Probeer het opnieuw.\n");
		}
	}

	printf("De oppervlakte van de rechthoek is %d\n", a * b);

}
```

5. Programmeer een while-loop die tot 10 kan tellen en daarna stopt. Je mag geen for-loop gebruiken.
6. Programmeer een loop die alle even getallen tussen 0 en 200 op de console print. Maak gebruik van een continue statement wanneer het een oneven getal betreft.
7. Maak een programma wat het gemiddelde van 6 cijfers (floats) kan bepalen. Bepaal of het gemiddelde hoger is dan een 5.5. Wanneer het hoger of gelijk is aan een 5.5 is print je het gemiddelde en voldoende. Wanneer het lager is dan een 5.5 print je het gemiddelde en onvoldoende.
8. Spaarsaldo. Je hebt een spaarrekening van 1000 euro. Op deze spaarrekening ontvang je 2.00% rente. Bereken voor 15 jaar wat het saldo van de spaarrekening is en print dit als een kolom per jaar op de console.
9. Fibonacci: De Fibonacci reeks bestaat uit een reeks integergetallen (zie tabel hieronder). Hier wordt het getal n bepaald door de twee vorige getallen bij elkaar op te tellen. Voorbeeld: index 2 : 0 + 1 = 1 en index 7: 5 + 8 =13

Index	0	1	2	3	4	5	6	7	8	9	10	11
Fibonacci	0	1	1	2	3	5	8	13	21	34	55	89

   Maak een programma wat de gebruiker vraagt om een index en wat het bijbehorende Fibonacci getal teruggeeft. Tip: gebruik een for loop.

10. Priemgetal: Een priemgetal is alleen deelbaar door zichzelf en door 1. Voorbeelden van priemgetallen zijn 2,3, 5, 7, 11, 13. Maak een programma dat kan bepalen of een getal een priem getal is. Tip: gebruik de % (modulo) operator om te controleren of een bepaalde deling een rest heeft en gebruik een for loop om alle delers te controleren.
11. Schrijf een programma wat een jaartal in Arabische (standaard) cijfers naar Romeinse cijfers kan omzetten tot het jaar 2000. Gegeven:
    I = 1, V = 5, X = 10, L = 50, C =100, D = 500, M = 1000. Verder hebben we:

Arabisch	4	6	7	8	9	40	60	70	80	90	1400	1600	1700	1800	1900
Romeins	IV	VI	VII	VIII	IX	XL	LX	LXX	LXXX	XC	MCD	MDC	MDCC	MDCCC	MCM

12. Maak 3 programma's. Vraag telkens de gebruiker om input voor n.
    a. Schrijf een programma wat faculteit van n (n!) kan bepalen (factorial) voorbeeld: $1! = 1, 2! = 1 \cdot 2 = 2,\ 3! = 1 \cdot 2 \cdot 3 = 6$
    b. Schrijf een programma wat het getal van euler benadert. $e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots \frac{1}{n!}$ Vraag de gebruiker om n.
    c. Schrijf een programma wat e^x kan bepalen. $e = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$. Vraag de gebruiker om n.
13. Programmeer de volgende tabel in een switch case statement wat het cijfer resultaat omzet in een beoordeling en deze op de console print.

1	Zwaar onvoldoende
2	Onvoldoende
3	Zeer matig
4	Matig
5	Bijna voldoende
6	Voldoende
7	Redelijk goed
8	Goed
9	Zeer goed
10	Uitmuntend
Ongeldig cijfer (niet 1 … 10)	Geen geldig cijfer

Week 4

Wanneer je vanaf deze week een functie schrijft, test deze dan door de functie meerdere keren aan te roepen met verschillende parameters. Print je resultaten op de console met een printf. Vergeet niet het prototype van de functie eerst te geven (voor de main) en de implementatie later. Bij de eerste paar functies geven we aan hoe je de parameters en return type kunt meenemen in het prototype. Bij de latere opdrachten moet je dit zelf kunnen bedenken.

1. double Gemiddelde(int a, int b, int c, int d): Schrijf een functie die het gemiddelde van 4 integers kan berekenen. De functie returnt een double en krijgt als argumenten 4 integers mee.
2. int AantalKeer(): Schrijf een functie die een integer n teruggeeft. Elke keer dat de functie wordt aangeroepen tel je een op. De integer n bevat dus hoe vaak die functie is aangeroepen.
3. void Swap(int* a, int* b): Gegeven zijn twee integers a en b. Schrijf een functie die deze twee integers kan omwisselen. Gebruik hiervoor het adres van beide variabelen.
4. int CijferSwap(int n): Schrijf een functie die een integer kan omdraaien. Bijvoorbeeld: 123 wordt 321.

Vanaf nu moet je zelf bepalen wat de parameters en return types zijn.

5. Som37(): Schrijf een functie die een integer n als parameter meekrijgt. Bereken de som van alle getallen deelbaar door 3 en 7 van 0 t/m n. Voorbeeld: 0 t/m 10, daar zijn 3, 6, 7, 9 deelbaar door 3 en 7. De som is dan 3+6+7+9=25.
6. KleinsteSomDeler(): 2520 is het kleinste getal wat deelbaar is door 1, 2, 3, 4, 5, 6, 7, 8, 9 en 10. Schrijf een functie die een integer n meekrijgt en het kleinste getal returnt wat deelbaar is van 1 t/m n. Voorbeeld voor n =4 krijg je 12 als het kleinste getal wat deelbaar is door 4, 3, 2 en 1.
7. IsPriem(): Schrijf een functie die kan bepalen of een getal een priemgetal is. Geef n mee als getal en return 1 wanneer dit het geval is en 0 wanneer dit niet het geval is.
8. PriemSom(): Schrijf een functie die alle priemgetallen tussen 1 en n bij elkaar op kan tellen. Gebruik hierin de som IsPriem().
9. Fibonacci(int index): Schrijf een functie fibonacci die zichzelf recursief aanroept. Index bepaalt het hoeveelste fibonacci getal. Je kunt dit doen door de volgende vergelijkingen in je Fibonaccifunctie te programmeren.

$$f_0 = 0$$
$$f_1 = 1$$
$$f_n = f_{n-1} + f_{n-2}, \text{ voor } n > 1$$

Week 5

We gaan eerst wat oefeningen doen met arrays. Daarna gaan we een wat groter programma schrijven, gebruikmakend van verschillende functies en arrays. Wanneer we een array willen meegeven aan een functie, let er dan op dat je het adres van de array en lengte meegeeft.

Gegeven zijn een aantal willekeurige arrays:

```
int a[10] = { 4,22,1,22,3,5,73,12,33,1 };
int b[10] = { 6,47,7,25,88,51,4,55,75,12 };
int c[10] = { 11,32,49,67,8,96,57,94,7,3 };
int d[10] = { 22,90,3,26,58,34,34,2,19,14 };
int e[10] = { 0 };
```

1. Schrijf een functie `void kwadrateer(int* arr, int length);` die een pointer naar en de lengte van de array meekrijgt. Vervang elk element van de array met het kwadraat ervan.
2. Schrijf een functie `int som(int* arr, int length);` die alle elementen van de array bij elkaar optelt en returnt.
3. Schrijf een functie `void optellen(int* arr1, int* arr2, int* arr3, int length);` die element voor element arr1 en arr2 kan optellen en opslaat in arr3.
   Dus arr3[2] = arr1[2] + arr2[2];
4. Schrijf een functie `int komtVoor(int* arr, int length , int getal);` die controleert of de int getal voorkomt in de array. Return 1 wanneer dit het geval is, return 0 wanneer dit niet het geval is.
5. Schrijf een functie `int maximum(int* arr, int length);` die het grootste getal van de array returnt.
6. Schrijf een functie `void sorteer(int* arr, int length);` die een array kan sorteren van klein naar groot.

Nu gaan we bezig met ons grotere programma. Maak hiervoor een nieuw project aan waarin je alle functies programmeert. Het doel is om een kalender van een willekeurig jaar na 2000 te kunnen printen. De opdrachten worden nu wel een stuk moeilijker, wanneer een opdracht niet lukt, probeer dan verder te gaan met de volgende opdracht. Je mag ook de opdrachten in twee etappes maken. Deel 1 (7 t/m 11) kun je deze week maken en Deel 2 (12 t/m 17) in week 6.

7. int IsSchrikkel(int y):

   Bepaal of een jaar een schrikkeljaar is. Je geeft hiervoor y (jaar) mee en returnt 1 wanneer het een schrikkeljaar is en 0 wanneer dit niet het geval is. Een schrikkeljaar is deelbaar door 4, maar niet door 100. Deelbaar door 400 is dan wel weer een schrikkeljaar. 1996 is een schrikkeljaar. 1900 is geen schrikkeljaar (deelbaar door 100, maar niet door 400). 2000 is een schrikkeljaar (deelbaar door 400).
   Test je code:

```
for (int i = 0; i < 2023; i++) {
            printf("%d: %d\n", i, isSchrikkel(i));
    }
```

8. int schrikkelDagen(int d1, int m1, int y1, int d2, int m2, int y2):

   Maak een functie die bepaald hoeveel schrikkeldagen er tussen 2 datums inzitten. Met een schrikkeldag bedoelen we hoe vaak de datum 29 februari tussen de twee data (d1-m1-y1 en d2-m2-y2) zitten. Geef daarvoor d1,m1,y1,d2,m2,y2 mee. Tip: zit je met m1 voor 1 maart is dan tel je het begin jaar eventueel mee (wanneer dit een schrikkeljaar is), en anders niet. Zit je met d2 na 28 februari dan tel je het laatste jaar eventueel mee.
   Test je code:

```
    printf("%d\n",schrikkelDagen(1, 1, 2000, 1, 2, 2000)); // 0
    printf("%d\n",schrikkelDagen(1, 1, 2000, 3, 3, 2020)); // 6
    printf("%d\n", schrikkelDagen(1, 1, 2000, 28, 2, 2020)); // 5
                                                  (tel 2020 niet mee)
    printf("%d\n", schrikkelDagen(1, 3, 2000, 28, 2, 2020)); // 4
                                               (tel 2000 en 2020 niet mee)
    printf("%d\n", schrikkelDagen(1, 3, 2000, 29, 2, 2020)); // 5
                                                  (tel 2000 niet mee)
    printf("%d\n", schrikkelDagen(1, 3, 2000, 30, 1, 2020)); // 4
    printf("%d\n", schrikkelDagen(1, 3, 2000, 28, 1, 2020)); // 4
```

9. int eersteWeekDagMaand(int m, int y):

   Schrijf een functie die 2 integers meekrijgt: m, y (maand en jaar). Bepaal de dag van de week waarop de maand begint en return dit van 0 t/m 6. (0: maandag, 1: dinsdag … 6: zondag). Tip: Begin bij 1 januari 2000 (dat was een zaterdag = 5). Je weet dat 1 februari dan op een dinsdag begint. Dat kun je bepalen door het aantal dagen ertussen op te tellen bij 5 en dan de modulo 7 te nemen: (5 + 31) % 7 = 1 = dinsdag. Zo weet je ook dat 2001 op een maandag begon, want (5+366) % 7 = 0.

Tip2: gebruik de functie aantal schrikkeldagen waar je het aantal schrikkeldagen gemakkelijk kunt bepalen tussen 1-1-2000 en een willekeurige datum.
Tip3: Om te weten hoeveel dagen er tussen 1 januari en 1 september zit moet je alle dagen van de maanden ertussen apart optellen. Gebruik hiervoor een static array met lengte 12 voor elke positie het aantal dagen in de maand. Omdat je slim gebruik kunt maken van de aantal schrikkeldagen functie en deze dagen apart kunt optellen, mag je er nu dus standaard van uit gaan dat februari 28 dagen heeft en een jaar 365 dagen.

Test je code:

```
printf("1 januari   2023: %d\n", eersteDagMaand(1, 2023));  // 6 (zondag)
printf("1 maart     2016: %d\n", eersteDagMaand(3, 2016));  // 1 (dinsdag)
printf("1 december  2002: %d\n", eersteDagMaand(12, 2002)); // 6 (zondag)
printf("1 juli      2004: %d\n", eersteDagMaand(7, 2004));  // 3 (donderdag)
printf("1 augustus  2019: %d\n", eersteDagMaand(8, 2019));  // 3 (donderdag)
printf("1 september 2023: %d\n", eersteDagMaand(9, 2023));  // 4 (maandag)
```

10. vulMaandArray(int m, int y, int* maand):

    We gaan een functie schrijven die een array kan vullen met alle datums van de maand. Hiermee bedoelen we dat je een array krijgt met [1,2,3,4,5…,30,31] voor bijvoorbeeld januari (data zijn 1 januari t/m 31 januari). In deze functie is m de maand is (1 t/m12), y is een jaar (2000 of hoger) en int* maand een pointer naar het adres van de array die we willen vullen met de data van die maand. Daarnaast gaan we de array aanvullen met nullen. Dit doen we zodat we straks de kalender mooi vorm kunnen geven. Maak eerst een array maand met lengte 42 en alleen maar nullen aan. Geef deze array (pointer) mee aan de functie vulMaandArray. De functie past onze array aan door deze met de dagen van de maand te vullen. De dagen die niet in de maand zitten behouden de waarde 0.
    Voorbeeld:
    april [1,2,3…29,30,0,0,0…],
    februari [1,2,3,…,27,28,0,0,0,…] of
    februari (schrikkeljaar) [1,2,3,…,27,28,29,0,0,…]

    Roep de functie isSchrikkel binnen je functie vulMaandArray(…) aan om te bepalen of je februari 28 of 29 dagen moet geven.

Uitgeprogrammeerd? Neem dan nu even pauze en ga in week 6 verder met de volgende opdrachten. Kun je er nog geen genoeg van krijgen, doe dan je best om de kalender werkend te krijgen!

11. vulMaandArray(int m, int y, int* maand):

    Pas vulMaandArray aan zodat deze nu een aantal nullen aan het begin van de array plaatst. Je bepaalt dit aantal nullen door de functie eersteWeekDagMaand (m,y) aan te roepen. Deze functie return de weekdag (0 t/m 6) waarop de maand begint. Wanneer dat op een donderdag is krijgen we 3 terug, dan willen we 3 nullen voor onze array plaatsen: 0, 0, 0, 1, 2 , 3, …, 30, 31, 0, 0, 0, 0, 0, 0, 0, 0 krijgen (3 voor de array, 8 aan het einde van de array). Onze oorspronkelijke array schuift dus met 3 plaatsen op.

12. printMaand(int* maand, int length):

    Schrijf een functie die de integer array (maandArray) van 42 waardes (length = 42) kan printen op de console. Plaats een spatie tussen elk cijfer.
13. Pas de functie aan zodat bij een waarde 0 twee spaties geprint wordt.
14. Pas de functie printMaand aan zodat de array per 7 waardes op een nieuwe regel geprint wordt.
    Bonus als je de cijfers mooi kunt uitlijnen (zie afbeelding van de kalender hieronder).
    Tip: je kunt hiervoor gebruikmaken van een extra spatie als je de getallen 1 t/m 9 moet printen.
    Bonus als je de afkorting van de dag erboven print (ma, di, …).

```
MA DI WO DO VR ZA ZO
      1  2  3  4  5
6  7  8  9  10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

15. printMaand(int* maand, int length, int maandnr):

    Pas de functie aan zodat je nu de maand erboven kan printen.
    Tip: gebruik hiervoor een `static char c[12][4]` met alle 3-letterige afkortingen van de maanden

```
        Jan
MA DI WO DO VR ZA ZO
      1  2  3  4  5
6  7  8  9  10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

16. Test nu vulMaandArray() en printArray() samen uit zodat je bijvoorbeeld de maand januari van 2021 kunt printen. Roep de functie printMaand als volgt aan:

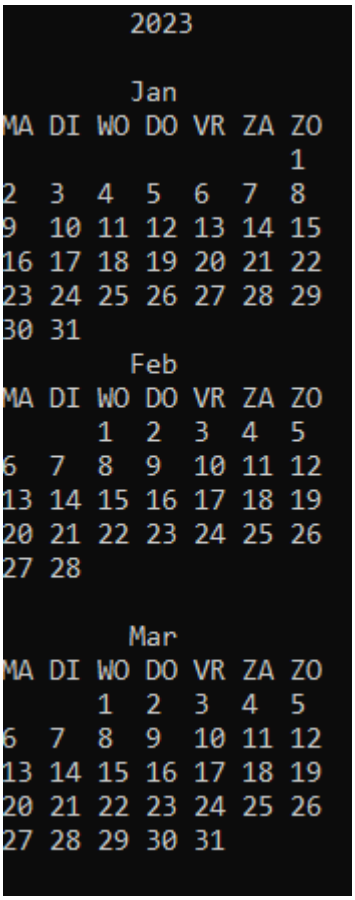
```
int huidigJaar = 2021;
int huidigeMaand = 1;
int maandArray[42] = { 0 };
vulMaandArray(huidigeMaand, huidgJaar, maandArray);
printMaand(2021, 42, 1);
```

    In dit voorbeeld beginnen we op positie 4 (tellen vanaf 0). Je gebruikt vulMaandArray(…) om eerst een array te vullen voor januari 2021. Wanneer je alle functies correct geprogrammeerd hebt, krijgt de array eerst 4 nullen en daarna de hele maand januari (1 t/m 31) gevolgd door 7 nullen. Daarna print je de array met printMaand(…). Deze print de array op de console.

17. MaandKalender(): Neem nu de code die je bij de vorige opdracht geschreven hebt en maak er een functie van. Maandkalender heeft als parameters m en y (maand en jaar). Dan roept hij zoals bij de vorige opdracht de juiste functies aan om uiteindelijk de kalender op de console te printen.

    Laat de maand beginnen op de juiste weekdag (4=vrijdag). Gebruik hiervoor de functie weekdag(1,1,y) //1 januari van dat jaar// Roep de functie isSchrikkel aan om te bedenken of het jaar een schrikkeljaar is of niet en laat hem de juiste hoeveelheid dagen printen afhankelijk van de maand en het schrikkeljaar.

18. JaarKalender(): Schrijf een functie waar je y, w (jaar, weekdag) meekrijgt. Deze functie print dan alle maandkalenders van dat jaar op de console:



    t/m december…

Week 6

Nadat je de opdrachten van deze week afhebt, kun je verder gaan met het programmeren van de kalender. Maak eerst deze opdrachten, zodat je tenminste weet hoe je bitwise manipulatie kunt toepassen.
Gebruik voor de volgende opdrachten `#include <stdint.h>` zodat je het datatype uint8_t kunt gebruiken (8 bits unsigned int)

1. Neem de volgende 8 variabelen over:
   uint8_t BIT0 = 0b00000001;
   uint8_t BIT1 = 0b00000010;
   uint8_t BIT2 = 0b00000100;
   uint8_t BIT3 = 0b00001000;
   uint8_t BIT4 = 0b00010000;
   uint8_t BIT5 = 0b00100000;
   uint8_t BIT6 = 0b01000000;
   uint8_t BIT7 = 0b10000000;
2. Wanneer je dit nog niet gedaan hebt, schrijf een functie void printBinary(uint8_t n) die een binairgetal n van 8 bits kan printen op de console.
3. Bepaal voor de volgende sommetjes de binaire uitkomst van de bitreeks links van het = teken. Schrijf deze eerst zelf op papier uit, en controleer ze dan in je programma:
   a. BIT0 = BIT0 & BIT1;
   b. BIT0 &= BIT1;
   c. BIT0 &= ~BIT1;
   d. BIT1 = BIT1 | BIT2;
   e. BIT1 |= BIT2;
   f. BIT1 |= BIT2 | BIT3 | BIT4 | BIT5 | BIT 6 |BIT 7;
   g. BIT1 ^= BIT4;
   h. BIT2 &= BIT3 | BIT4| BIT 5 | BIT 6;
4. Maak 3 variabelen aan van het type uint8_t a,b en c. Geef deze de waarden 0b1001000, 0b11110011 en 0b1100011. Gebruik deze variabelen om de functies in de volgende opdrachten mee te testen.
5. Schrijf 3 functies:
   a. uint8_t BITOR(uint8_t a, uint8_t b) – deze returnt de or van de twee bits (|=)
   b. uint8_t BITANDINV(uint8_t a, uint8_t b) – deze returnt de and van bit a en de inverse bit b. (&=~)
   c. uint8_t BITTOGGLE(uint8_t a, uint8_t b) – deze XOR't bit in reeks a met b. (^=). Dus daarmee kun je bits togglen.
6. Demonstreer de werking van de functies m.b.v.van een aantal testen met BIT0 t/m BIT7.
7. Schrijf een functie uint8_t BITSHIFT (uint8_t a, int b) die bitreeks a met b stappen naar rechts schuift. Als b negatief is schuif je met -b stappen naar links
8. Maak een eigen typdef enum dag d variabele waarbij je enum dag alle dagen van de week geeft. Schrijf een bijbehorend switch case statement waarin je switcht over de verschillende dagen. Print bij elke mogelijkheid op de console de dag. Test daarna je code Voorbeeld:
   case ma:

   ```
   printf("maandag\n");
   break; //…
   ```