# Opgave 1

*Skriv en rekursiv algoritme, som har et naturligt tal som parameter og returnerer summen af de ulige tals kvadrater fra 1 til N.*

*Eksempel: kaldt med parameteren 8 returneres*

*Det er vigtigt at optimere algoritmen, så overflødige rekursive kald undgås.*

**Svar**

A computer screen shot of a program code

AI-generated content may be incorrect.

Jeg starter med at lave et string , dette variable vil lave den endelige formular. Int får resultatet fra rekurasiv algoritme af 8, som er 84. I funktionen, tjekker jeg først om n er mindre eller i ligemed 0. Hvis n er støre, så tjekker jeg om tallet er ulige. Hvis tallet er ulige, så tilføjer jeg til den nuværende formular, og derefter retunerer

# Opgave 2

Hvad er Store-0 tidskompleksiteten for nedenstående algoritme. Begrund dit svar.

A white background with blue text

AI-generated content may be incorrect.

**Svar**

***Trin 1: Øverste loop:***

her bliver **i** dobblet med **i,** for hver iteration. Derfor kommer vi frem til )

***Trin 2: Midten loop:***

Her sker der det samme. Hvor bliver dobbelt hver gang. Derfor får vi

***Trin 3: Inner loop:***

Den indre loop kører gange, da den er afhængig af og kvadratrod af .Derfor får vi

***Trin 4: Kombination***

*Vi ganger alle tidskompleksiteter* *sammen*

***Trin 5: Sidste loop***

Her er det simpel notation, da vi har

***Trin 6: Total tidskompleksiteten***

Den dominerende term er , da den gror hurtigere end

Derfor er *tidskompleksiteten*

A graph of a function

AI-generated content may be incorrect.

# Opgave 3

Skriv en *rekursiv* algoritme med følgende signatur:

bool additive(String s)

Parameteren indeholder en streng bestående af cifre, fx. “82842605”.

Algoritmen returnerer true, hvis parameteren indeholder en substring af tre på hinanden efterfølgende tal, hvor det tredje ciffer er lig med summen af de to forrige.

I ovenstående eksempel returneres *true,* fordi indeks 5 (6) er summen af indeks 3 og 4 (4 plus 2). Tip: ASCll-værdien af karakteren ‘7’ er 55.

**Svar**

A screen shot of a computer code

AI-generated content may be incorrect. betyder at vi udnytter ASCII-værdierne. Så hvis 7 har ASCII værdien på 55 og vi subtraherer med ASCII værdien på 0, får vi , som er det samme som 7.

# Opgave 4

Skriv en algoritme, der har et array af usorterede, entydige naturlige tal som input og find de tre tal i arrayet, hvis sum er tættest på en potens af 2. Det samme tal kan kun bruges en gang.

Algoritmens returværdi skal være et heltalsarray, som først indeholder de tre tal og dernæst den tilhørende potens af to (fx 512).

Kaldt med arrayet (23,56,22,11,65,89,3,44,87,910,45,35,98}, returneres de tre tal 89, 3, 35 og potensen af 2: 128.

Hvad er Store-O tidskompleksiteten af din algoritme? Begrund dit svar og diskuter mulighederne for at optimere din løsning yderligere.

**SVAR:**

***Trin 1: Øverste loop:*** for (int i = 0; i < numbers.Length - 2; i++)

Loop kører

***Trin 2: Midten loop:*** for (int j = i + 1; j < numbers.Length - 1; j++)

Loop kører

***Trin 3: Inner loop:*** for (int k = j + 1; k < numbers.Length; k++)

Loop kører

***Trin 4: Kombination***

Vi ganger alle tidskompleksiteter sammen

***Trin 5: Total tidskompleksiteten***

A graph with a line drawn on it

AI-generated content may be incorrect.

Den nuværende tidskompleksitet kan forbedres til , ved at ikke anvende 3 nestede loops. Det kan man gøre ved at kun anvende 2 nestede loops, og derefter tilføje den trejdre værdi ind til den indre loop.

A screen shot of a computer screen

AI-generated content may be incorrect.

# Opgave 5

Hvad er Store-0 tidskompleksiteten af nedenstående metode? Begrund dit svar.

A white background with black text

AI-generated content may be incorrect.

***Trin 1: Øverste loop:***

Loop kører gange:

***Trin 2: Midten loop:***

Loop kører N gange for hver i:

***Trin 3: Inner loop:***

K bliver ganget med 2, hver iteration:

***Trin 4: Kombination***

Vi ganger alle tidskompleksiteter sammen

***Trin 5: Total tidskompleksiteten***

A graph on a graph paper

AI-generated content may be incorrect.

# Opgave 6

Skriv en rekursiv algoritme med følgende signatur:

int sumDivisibleBy3(int N)

Algoritmen returnerer summen af heltal større end 0 og mindre end eller lig med N, som er dividér bare med 3.

Kaldt med N = 12 er den korrekte returværdi 30 (3+6+9+12). Kaldt med N = 14 er den korrekte returværdi også 30.

Din algoritme skal optimeres således, at overfledige rekursive kald undgås.

**SVAR:**

A screenshot of a computer program

AI-generated content may be incorrect.

Vores endlige O notation bliver til O(N)

# Opgave 7

6561 er et eksempel på et naturligt tal, som kan skrives som hvor X og Y er heltal, dvs. 94. Et andet eksempel er 3125 ().

Skriv en algoritme, som kan afgøre om et givent naturligt tal Z < 100,000 (algoritmens parameter) kan skrives som hvor X og Y er heltal, hvorom det gælder, at X > 2 og Y > 2.

Algoritmens returværdi skal designes således, at felgende information kan udledes fra den:

* Værdien af X (i tilfældet 6561 er X=9).
* Værdien af Y (i tilfældet 6561 er Y=4).
* For det givne Z kan der være flere Iøsningspar (X,Y). For 3125 er der kun et Iøsningspar; men for 6561 er X=3 og Y=8 også en mulighed. Hvis der er mere end en løsning, skal den med den største X-værdi returneres.
* Hvis det givne Z ikke har noget lesningspar (X,Y) eller Z indeholder en ulovlig værdi, returneres en dummy eller default værdi.

Algoritmen skal optimeres under antagelsen af, at maksimumværdien for Z og minimumsværdierne for X og Y aldrig ændrer sig.

**SVAR:**

Funktionen er ret simpel. Først definere jeg to variabler for X og Y som begge er -1. Derefter kører jeg en for loop, hvor der for hvert iteration bliver lavet en ny x værdi. For example . Derefter tjekker if statement, hvis , og så ændre værdierne for bestX og bestY. Til sidst retuneres værdierne bestX og bestY

A screen shot of a computer program

AI-generated content may be incorrect.

# Opgave 8

Tabellen nedenfor er en hash tabel, hvor der anvendes quadratic probing til collision resolution.

Indeks Værdi

|  |  |
| --- | --- |
| 0 |  |
| 1 |  |
| 2 | V |
| 3 | R |
| 4 |  |
| S |  |
| 6 | P |
| 7 |  |
| *8* | E |
| 9 |  |
| 10 | F |

Indeks 0, 1, 4, 5, 7 og 9 er ledige. Vis hvordan tabellen ser ud, efter at elementerne Q C and H, som hasher til henholdsvis indeks 7, 8 and 2 er indsat. Begrund dit svar.

**SVAR:**

Jeg har lavet et algoritme i C#, som anvender quadratic probing. Hvis Q skal hashe til indeks 7, så kan vi bare indsætte den, så pladsen er tom. C har ikke plads i række 8, og derfor udnytter vi quadratic probing collision resolution. , dvs C skal på række 9. H har ikke plads på række 2, igen bruger vi probing. . H skal på række 0.

Indeks Værdi

|  |  |
| --- | --- |
| 0 | H |
| 1 |  |
| 2 | V |
| 3 | R |
| 4 |  |
| S | Q |
| 6 | P |
| 7 |  |
| *8* | E |
| 9 | C |
| 10 | F |

Nedunder er lavet kode, hvis man bruger ASCII værdierne for indsætning af bogstaverne.

A black rectangular object with white text

AI-generated content may be incorrect.A screen shot of a computer code

AI-generated content may be incorrect.

# Opgave 9

Hvad er Store-O tidskompleksiteten for nedenstående metode? Dit svar må godt være baseret på udførte tests. Hvis du vælger at give et svar uden tests, skal svaret begrundes.

A white background with blue text

AI-generated content may be incorrect.

**SVAR:**

O tidskompleksiteten ender med at være *,* fordi algoritmen køre igennem de samme resultater mange gange, som betyder den gror exponentielt. Man kan tænke på den som et træ. Den føste gange man køre funktionen, splitter den i to andre funktioner, som derefter bliver splittet igen og igen, indtil de når

A screen shot of a computer screen

AI-generated content may be incorrect.

A black background with white numbers and a black rectangle

AI-generated content may be incorrect.

A graph with a green line

AI-generated content may be incorrect.

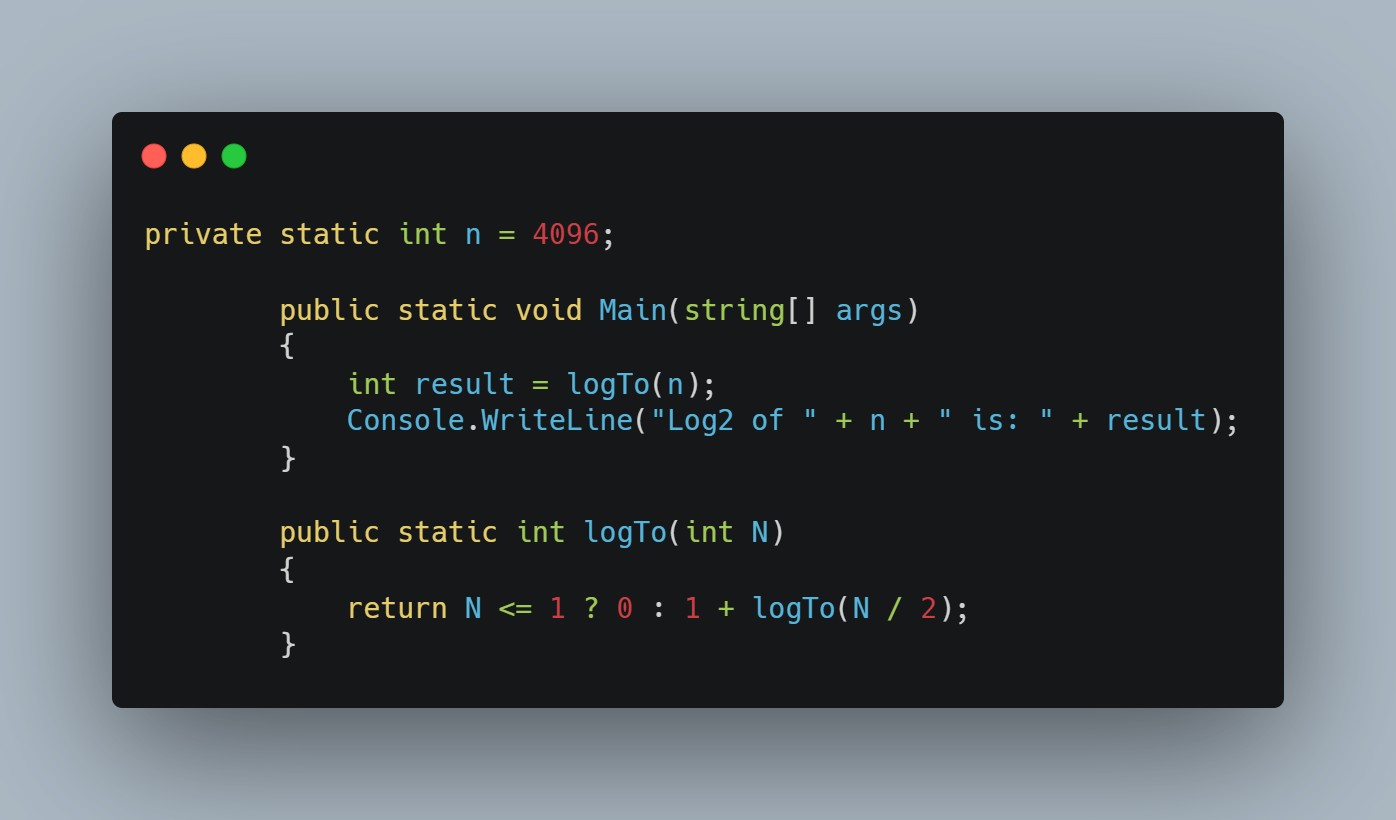
# Opgave 10

Skriv en rekursiv metode med følgende signatur:

Algoritmen returner totals-logaritmen af N, og det er en forudsætning, at N er et naturligt tal og en potens af 2.

Kaldt med returneres 5, og med returneres .

**SVAR:**



Her er der lavet en meget simpel funktion, som klarer opgaven. Funktionen bruger en one-line if statement som tjekker, om N er mindre end eller lig med 1. Hvis det er tilfældet, returneres 0, ellers returneres For hvert rekursivt kald deles N med 2, og der lægges 1 til resultatet. Det betyder, at funktionen tæller, hvor mange gange N kan halveres, indtil det bliver 1. For eksempel giver resultatet 12, fordi 4096 kan halveres 12 gange før det når 1.

# Opgave 11

Tabellen nedenfor repræsenterer de afgivne stemmer ved et valg.

I dette eksempel er der 7 kandidater (1-7), og der er afgivet 12 stemmer. Kandidat 6 fik 1 stemme, kandidaterne 1, 3, 4 og 7 fik hver 2 stemmer, kandidat 5 fik 3 stemmer, og kandidat 2 fik 0 stemmer.

Opgaven går ud på at skrive en algoritme, som kaldt med tabellen og eventuelt tabellens længde, kan afgøre om en kandidat fik mere end 50 % af stemmerne. I så fald returneres kandidatens nummer. Hvis ingen kandidat fik over 50 % af stemmerne, returneres -1.

I eksemplet opnåede ingen af kandidaterne flertal, og der returneres -1. Hvad er din algoritmes tidskompleksitet?

**SVAR:**

Funktionen har kun en loop som kører N gange. Dvs O- tidskompleksitet er

