1. **Jednoduché a složené (strukturované) datové typy.**
   Proměnná, konstanta. Jednoduché datové typy (int, char, bool, double, signed x unsigned ...) – velikosti, rozsah. Složené (strukturované) datové typy: pole, struktura, textové řetězce. Ukazatele. NULL. Rozdíl mezi interpretovaným a překládaným jazykem. Hodnotové a referenční datové typy.

2. **Ukazatele. Spojové datové struktury. Pole.**
   Adresování (adresa proměnné). Ukazatelé v C. Datový typ ukazatele. Referenční a dereferenční operátor (&,). Příklad využití. NULL. Pointerová aritmetika. Sizeof. Časté chyby při práci s pointery. Spojové datové struktury (spojový seznam). Časová složitost základní operací. Nakreslit ilustrační obrázky. Souvislost s ukazateli. Pole. Co to je, k čemu to je. Předávání pole v parametru funkce (souvislost s ukazateli). Srovnání výhod a nevýhod spojových datových struktur a polí.

3. **Fronta a zásobník.**
   Fronta: co to je (*datová struktura*), demonstrace na obrázku, základní metody (*Enqueue*, *Dequeue*). Zásobník: co to je, demonstrace na obrázku, základní metody (*Push*, *Pop*). Časová náročnost operací vzhledem k implementaci (pole, spojový seznam). Reprezentace v jazyce C# (popř. v C). Příklady algoritmů, kde se tyto struktury používají (daný algoritmus umět popsat). Souvislost rekurze se zásobníkem.

4. **Podprogramy.**
   Metody. Funkce. Klíčové pojmy: návratová hodnota, parametr. Předávání parametrů hodnotou a odkazem. Vhodná i nevhodná použití. Lokální a globální proměnné. Základní řídící programové struktury – větvení a cykly. Zásady správného pojmenování funkce/metody.

5. **Algoritmus a jeho vlastnosti.**
   Pojem algoritmus, vlastnosti. Porovnání algoritmů z hlediska asymptotické časové a prostorové složitosti. Příklady algoritmů fungujících v konstantním, lineárním, kvadratickém a logaritmickém čase.

6. **Rekurze a její využití.**
   Pojem rekurze. Příklady využití (např. Fibonacci, permutace, faktoriál). Uveďte časovou složitost daného algoritmu. Výhody a nevýhody rekurze. Kde je (ne)efektivní ji použít. Koncová rekurze. Jak lze rekurzi nahradit použitím zásobníku. *StackOverflow exception*. Navazující témata: DFS, Backtracking, Merge sort, Quick sort ...

7. **Statická a dynamická alokace paměti. Práce s textovými soubory.**
   Statická x dynamická alokace paměti, *malloc()*, *free()*). Halda a zásobník. Garbage collector v C#. Práce s textovými soubory v C#. StreamReader, StreamWriter – klíčové funkce a metody. Blok *using*. Časté chyby při práci se soubory.

8. **Časová a paměťová složitost.**
   Vysvětlení pojmu – co, k čemu, proč. Nejhorší, nejlepší a průměrný případ. O-notace. Způsob určení časové a prostorové složitosti. Vzhledem k čemu časovou složitost určujeme. Příklady algoritmů fungujících v konstantním, lineárním, kvadratickém a logaritmickém čase. Vylepšení exponenciální časové složitosti při výpočtu Fibonacciho čísel.

9. **Reprezentace grafu v počítači.**
   Definice grafu. Matice sousednosti. Matice incidence. Seznamy sousedů. Časová náročnost základních metod těchto reprezentací. Která reprezentace se hodí na jaký typ grafu. Reprezentace grafu pomocí objektově orientovaného programování.

10. **Stromy a jejich využití. Průchod stromem.**
    Definice stromu. Definice binárního stromu. Definice binárního vyhledávacího stromu. Algoritmus procházení libovolného stromu. Algoritmus hledání prvku v BVS. Průchod stromem do hloubky i do šířky. Co může být ve stromu uloženo. Co je halda a k čemu slouží. Příklady využití stromů. Možný způsob implementace.

11. **Insertion sort. Selection sort.**
    Motivace pro třídění dat. Popište po jednotlivých krocích oba algoritmy. Znázorněte na obrázku. Časová a paměťová složitost.

12. **Bubble sort. Merge sort.**
    Motivace pro třídění dat. Popište po jednotlivých krocích oba algoritmy. Znázorněte na obrázku. Časová a paměťová složitost.

13. **Quick sort.**
    Motivace pro třídění dat. Popište po jednotlivých krocích. Volba „dobrého“ pivota. Využití v praxi. Znázorněte na obrázku. Časová a paměťová složitost.

14. **Heap sort.**
Motivace pro třídění dat. Definice haldy. Popište algoritmus po jednotlivých krocích. Znázorněte na obrázku. Časová a paměťová složitost.

15. **Lineární a binární vyhledávání. Vyhledávací stromy.**
Srovnání vyhledávání v nesetříděném poli, setříděném poli, binárním vyhledávacím stromu. Srovnání lineárního a binárního vyhledávání z hlediska časové složitosti. Příklady ze života, kdy které používáme. Definice binárního stromu. Definice binárního vyhledávacího stromu. Základní operace (algoritmy): hledání daného klíče, hledání minima, vkládání a mazání. Časová složitost těchto operací. Vyváženost stromu. Zmínka o AVL stromech.

16. **Rozděl a panuj. Dynamické programování. Backtracking.**
Klíčová myšlenka *Rozděl a panuj*, ukázka na příkladu. Hlavní myšlenka dynamického programování, rozdíl oproti *Rozděl a panuj* příklad. Hlavní myšlenka *backtrackingu*, příklad využití. Pro jaké typy úloh použijeme který přístup.

17. **Aritmetické výrazy – reprezentace v grafu, vyhodnocení.**
Různé reprezentace aritmetických výrazů: infix, postfix a binární strom. Algoritmy vyhodnocení výrazů ve všech zmíněných reprezentacích. Algoritmy převodu výrazů z binárního stromu na infix, prefix, postfix a naopak.

18. **Objektově orientované programování.**
Základní myšlenky OOP, užitečnost. Vysvětlete na konkrétních příkladech následující pojmy: třída, instance/objekt, dědičnost, polymorfismus, zapouzdření, vlastnost, metoda, konstruktor. Zapouzdření. Abstraktní třída. Rozhraní.

19. **Kolekce .NET.**
Využití kolekcí, výhody. List. Dictionary (hashování). Queue. Stack. SortedList. Generické kolekce. Porovnání z hlediska časové složitosti základních operací.

20. **Programování řízené událostmi. Okenní aplikace.**
Základní princip fungování interaktivních aplikací, role událostí. Průběh zpracování události. Příklady základních událostí při vytváření okenní aplikace. Základní pravidla při vytváření a programování okenních aplikací. Responzivní vzhled aplikace.

21. **Základy teorie grafů. Bipartitní graf.**
Definice pojmů: graf, ohodnocený graf, orientovaný graf, souvislý graf, cesta, sled, nejkratší cesta, cyklus/kružnice, strom, komponenta souvislosti, úplný graf. – vše ukažte na příkladech. Bipartitní graf. K čemu se používá. Problém největšího párování. Popište reprezentace grafu v počítači.

22. **Prohledávání do hloubky a do šířky.**
Jednotlivé kroky obou algoritmů. Omezení. Fronta. Časová složitost. Příklady úloh vedoucí na použití těchto algoritmů. Souvislost s hledáním nejkratší cesty.

23. **Hledání minimální kostry grafu.**
Definice pojmu kostra grafu, minimální kostra grafu. Motivační příklad. Zvolte si jeden z algoritmů: *Jarník*, *Borůvka*, *Kruskal* a ten popište (ideálně na obrázku).

24. **Topologické třídění a jeho využití.**
Motivace. Pojmy: cyklus v grafu, acyklické orientované grafy, detekce cyklů, topologické uspořádání vrcholů grafu (ukažte na obrázku). Popište algoritmus topologického třídění grafu. Proč tento algoritmus funguje. Využití DFS. Časová složitost algoritmu. Využití topologického třídění.

25. **Hledání nejkratší cesty v grafu.**
Definice pojmů: graf, ohodnocený graf, vzdálenost, cesta, nejkratší cesta, záporné hrany. Motivační příklad z praxe k využití algoritmů pro hledání nejkratší cesty. Příklady úloh vedoucích na hledání nejkratší cesty v grafu. Využití BFS, DFS. Dijkstrův algoritmus s minimovou haldou, omezení.