Fragekatalog zur ersten AP-Schulaufgabe

1. Was versteht man unter einer Referenz in C++
2. Erklären Sie die Unterschiede zwischen einem C++ Pointer und einer C++ Referenz
3. Weshalb dürfen Referenzen nicht auf NULL gesetzt werden?
4. Erklären Sie auf welche Weise Funktionen in C++ „indirekt" mehrere Werte gleichzeitig zurückgeben können!
5. Was versteht man in C++ unter einem Funktionstemplate?
6. Erläutern Sie Probleme bei der Nutzung von Fuunktionstemplates!
7. Beschreiben Sie die Vorteile von C++ Fuunktionstemplates!
8. Erstellen Sie ein Funktionstemplate zu folgender Problembeschreibung...
9. Aus welchen drei Teilen besteht die ANSI C++ Standard Template Library?
10. Erklären Sie den Zweck von STL Algorithmen / Containern / Iteratoren
11. Nennen Sie die Arten von Containern, welche in der STL implementiert sind!
12. Beschreiben Sie die grundsätzliche Funktion der in C++ implementierten Container List, Set, Map, Queue und / oder Stack
13. Definieren Sie z.B. einen list / vector – Container, der Zeiger auf double Werte speichern kann! Fügen Sie dem Container zwei Werte am Anfang hinzu!
14. Lesen Sie alle Werte aus einem Container über einen Iterator aus!
15. Erläutern Sie die Begriffe Kapselung, Verbergen von Daten, Vererbung und Polymorphie anhand eines Beispiels!
16. Erklären Sie den Unterschied zwischen Klasse und Objekt!
17. Was versteht man unter

a) einem Attribut?
b) einer Methode?
c) einer Instanz?

18. Welche Eigenschaften und Methoden kann z.B. eine Klasse Bankkonto besitzen? (je drei Eigenschaften und Methoden)
19. Nennen Sie vier Zyklen eines objektorientierten Projekts!
20. Beschreiben Sie den Ablauf eines objektorientiert durchgeführten Projekts
21. Beschreiben Sie die Prozesse bei der Analyse eines objektorientierten Projekts
22. Erklären Sie den Unterschied zwischen Basis-, statischem und dynamischen Modell beim Design eines objektorientierten Projekts!
23. Erläutern Sie die Prozesse bei der Realisierung eines objektorientierten Projekts!
24. Beschreiben Sie die Abläufe beim Einsatz eines objektorientierten Projekts
25. Erklären Sie, weshalb die Zyklen bei der OOP mehrmals durchlaufen werden
26. Welche Vorteile bringt das Spiralmodell mit sich?

27. Warum wird bei der OOP auf das Design besonderes Augenmerk gelegt
28. Vergleichen Sie den Entwurf eines strukturierten Softwareprojekts mit dem eines objektorientierten Objekts
29. Was versteht man unter einem Anwendungsfall?
30. Beschreiben Sie die Symbolik des Anwendungsfallmodells!
31. Erläutern Sie die Bedeutung von Akteuren!
32. Wozu benötigt man die <<include>>-Beziehung bei Anwendungsfallmodellen
33. Beschreiben Sie den Unterschied zwischen der <<include>>- und der <<extend>>-Beziehung!
34. Erklären Sie den Begriff „extension point"!
35. Vergleichen Sie das Konzept der Vererbung bei Klassen, mit dem bei Anwendungsfällen
36. Welche Vorteile bietet die Darstellung von Anwendungsfällen beim Softwareentwurf?
37. Erstellen Sie ein Anwendungsfallmodell für folgende Problembeschreibung …
38. Erläutern Sie die Aussage des folgenden Klassendiagramms / Objektdiagramms nach UML! …
39. Beschreiben Sie die Beziehung zwischen Klassen, die durch Vererbung entsteht
40. Was versteht man unter einer abstrakten Klasse?
41. Erklären Sie den Unterschied zwischen einer Interface-Klasse und einer abstrakten Klasse
42. Wie werden abstrakte Klassen/ Interface Klassen in C++ erstellt
43. Nennen Sie … mögliche Assoziationen aus dem objektorientierten Entwurf und geben Sie je ein Beispiel dazu an!
44. Erklären Sie den Unterschied zwischen einer Aggregation und einer Komposition!
45. Was versteht man unter einer Abhängigkeit?
46. Erläutern Sie die <<use>>-Abhängigkeit zwischen zwei Klassen anhand eines Beispiels!
47. Beschreiben Sie die Rollen von Client und ein Supplier bei der Abhängigkeitsbeziehung
48. Was versteht man unter einer Assoziationsklasse?
49. Zeichnen Sie ein Klassendiagramm / Objektdiagramm nach UML, das die folgende Situation beschreibt …
50. Erläutern Sie die Unterschiede der Schutzattribute public, protected und private!
51. Implementieren Sie die Klassendefinitionen in C++ zu folgenden Klassendiagrammen nach UML …
52. Welchem Zweck dient ein Zustandsdiagramm / Aktivitätsdiagramm / Kommunikationsdiagramm / Sequenzdiagramm / Timingdiagramm?
53. Weshalb verwendet man in der objektorientierten Programmierung keine Struktogramme zur Darstellung des Programmablaufs?
54. Worin unterscheiden sich Kommunikationsdiagramme von Sequenzdiagrammen?
55. Was versteht man unter dem „Tokenprinzip" bei Aktivitätsdiagrammen?

56. Erläutern Sie das Konzept der Partitionen beim Aktivitätsdiagramm
57. Welche Diagramme der UML zählen zu den Interaktionsdiagrammen?
58. Unter welchen Umständen setzt man Interaktionsübersichtsdiagramme in der UML ein?
59. Erläutern Sie die Bedeutung der dynamischen Modellierung für die objektorientierte Programmentwicklung!
60. Erstellen Sie ein Zustandsdiagramm / Aktivitätsdiagramm / Kommunikationsdiagramm / Sequenzdiagramm zu folgender Problembeschreibung …
61. Erstellen Sie den C++ Quelltext zu folgendem Zustandsdiagramm / Aktivitätsdiagramm / Kollaborationsdiagramm / Sequenzdiagramm zu folgender Beschreibung …
62. Nennen Sie die beiden UML Modellelemente des Komponentendiagramms
63. Erklären Sie den Unterschied zwischen einer Komponente und einem Artefakt!
64. Erläutern Sie drei gebräuchliche Stereotypen von Artefakten!
65. Was versteht man unter einem „Subsystem“?
66. Auf welche Weise werden Schnittstellen symbolisiert?
67. Erläutern Sie die Bedeutung von Komponenten hinsichtlich der Wiederverwendbarkeit von Software!
68. Erstellen Sie ein Komponentendiagramm / Verteilungsdiagramm zu folgender Problembeschreibung …

Viel Erfolg beim Lernen!