Informations générales

Examen de module	Examen écrit	Date : 17.03.2022
Nom du module : **Programmation orientée objet**	Numéro de module :	40050200
Durée de l'examen : **60 min + 15 min**	Examinateur : Welp	

Nom : _____ Prénom_____

Numéro de matricule : _____

Évaluation

Tâche	1	2	3	4						Total
Points à atteindre	9	12	8	18						47
Points obtenus										
Signature de l'examinateur				Points obtenus en					Note	

Tâche 1: Du C au C++

1. Ecrivez une fonction `résistance totale` pour calculer la résistance totale de deux résistances électriques montées en parallèle ou en série. La fonction doit recevoir comme paramètres les deux valeurs de résistance `r1` et `r2` ainsi qu'un identificateur `k` de type `int`, qui indique si la résistance totale doit être calculée pour un montage en parallèle (`k=0`) ou pour un montage en série (`k=1`). L'indication de l'identificateur est facultative. S'il est omis, le calcul doit porter sur un montage en série.

   Résolvez ce problème en

   1. `résistance totale` surchargée
   2. Définir la `résistance totale` comme une fonction avec des paramètres par défaut.

   Donnez les définitions des **fonctions** pour les deux solutions. (6 points)

2. Qu'est-ce que le fragment de programme suivant affiche à l'écran ? (3 points)

```
long n[3] = {300,200,100};
long& r = n[0];
long* p = &n[2];
n[1] = r/2;
p=p-1;
r = *p + n[0];
cout << n[0] << "  " << n[1] << "  " << r << endl;
```

Tâche 2: Classes et objets

Définissez une classe `Cercle` pour la représentation de cercles. La classe doit disposer des propriétés suivantes *(NB : laissez suffisamment de place dans la classe pour pouvoir faire des ajouts au fur et à mesure)* :

- Le rayon du cercle doit être enregistré comme attribut dans la classe. Il ne doit pas être possible d'accéder directement au rayon d'un cercle. L'accès doit se faire via les méthodes setter et getter correspondantes. **Définissez (implémentez)** les méthodes comme des fonctions en ligne. Les rayons négatifs ne doivent pas être possibles.
- La classe doit disposer de constricteurs appropriés pour l'initialisation d'`objets circulaires`. Un `objet cercle` doit pouvoir être initialisé en indiquant le rayon. Il doit également être possible d'initialiser un objet sans paramètre. Dans ce cas, l'objet doit être initialisé avec un rayon de 1.0 (*voir remarque*). **Définissez** les méthodes comme des fonctions en ligne.
- La classe doit fournir une méthode `getUmfumfang()` qui renvoie la circonférence du cercle. **Définissez** la fonction en dehors de la classe.
- La classe doit disposer d'un attribut de classe dans lequel le nombre d'objets `circulaires` est enregistré (compteur d'objets). Le nombre d'`objets circulaires` doit pouvoir être interrogé via une méthode de classe. **Définissez** l'attribut et la méthode.
- En outre, l'**`opérateur /`** à deux chiffres doit être surchargé pour la classe `Cercle`. Celui-ci doit permettre la "division" d'un cercle par une valeur `x`. Une expression correspondante doit fournir un cercle dont la circonférence est `1/x fois plus` grande. **Déclarez** l'opérateur. *(voir note)* (12 points)

Remarque

La classe `Cercle` doit par exemple pouvoir être utilisée de la manière suivante

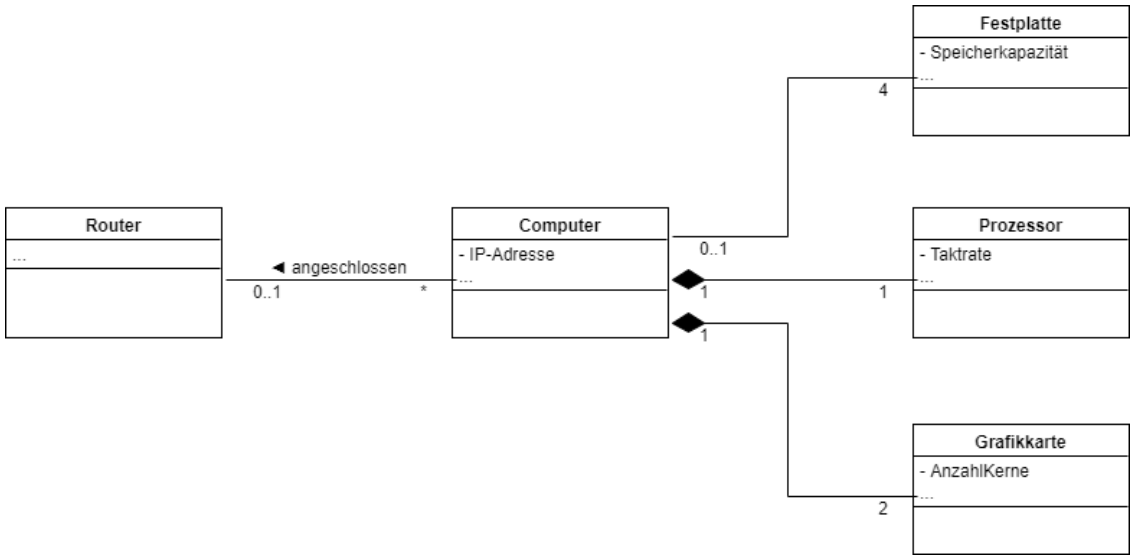
```
int main()
{
    Cercle k1 ;
    Cercle k2(3.0)
    ;
    Cercle k3 ;

    cout << "cercle avec rayon " << k1.getRadius() << " a la circonférence " << k1.getUmfang() <<
    endl ; cout << "cercle avec rayon " << k2.getRadius() << " a la circonférence " <<
    k2.getUmfang() << endl ;

    k3=k2/3 ;
    cout << "k2/3 a un rayon " << k3.getRadius() << " et une circonférence " << k3.getUmf Umfang() <<
    endl ;

    cout << "Il y a un total de " << Cercle::getZaehler() <<" cercles" << endl ;
}
```

édition :

```
Le cercle de rayon 1 a une
circonférence de 6.283 Le cercle de
rayon 3 a une circonférence de 18.849
k/3 a un rayon de 1 et une
circonférence de 6.283
Il y a en tout 3 cercles
```

Tâche 3 : Relations entre objets et modèles

Dans un logiciel de gestion de réseau, il y a les classes `routeur`, `ordinateur`, `disque dur`, `processeur` et `carte graphique`, qui sont en relation les unes avec les autres selon le diagramme de classes suivant.



1. Quels attributs sont nécessaires dans la classe `Ordinateur` pour implémenter les relations avec les classes `Routeur`, `Disque dur`, `Processeur` et `Carte graphique` ? Répondez à la question en donnant la définition de classe pour la classe Ordinateur (attributs uniquement, les méthodes ne sont pas nécessaires). (4 points)

2. **Définissez** une **fonction de** modèle `min qui` renvoie la plus petite de deux valeurs. Les deux valeurs doivent être de n'importe quel type. (4 pts)

   application possible :

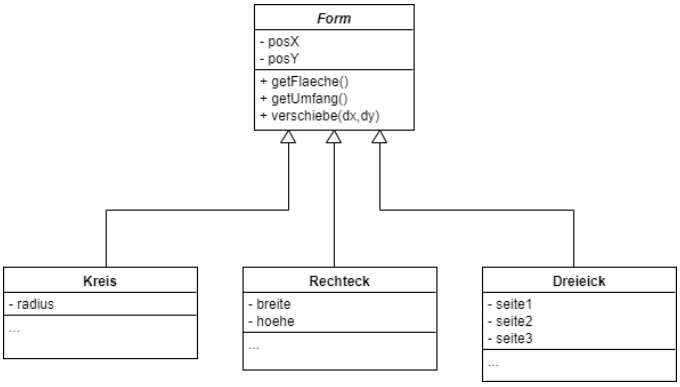
   ```
   cout << min(7,2) << endl ;
   cout << min("Bochum", "Dortmund") << endl ;
   ```

   édition :

   ```
   2
   Bochum
   ```

Tâche 4: Héritage et polymorphisme

Soit le diagramme de classes suivant pour des formes géométriques :



La classe `Form` est définie comme suit :

```
class Form
{
    public:
        Form(int x, int y);
        void verschiebe(int dx, int dy);
        virtual double getFlaeche()=0;
        virtual double getUmfang()=0;
    private:
        int posX,posY;
};
```

1. La classe `Forme` est une classe **abstraite**. (3 points)
   a. Comment devient-elle une classe abstraite ?
   b. Qu'est-ce que cela signifie pour l'utilisabilité de la classe ?
2. Dérivez une classe concrète `Rectangle` à partir de la classe `Forme`. Donnez la définition de la classe, y compris les définitions des méthodes. (5 points)
3. De quels attributs disposent les objets de la classe `Rectangle` ? (2 points)
4. Ecrivez une fonction `surfaceTotal` pour calculer la surface totale d'un champ de formes géométriques (cercles, rectangles, triangles). (6 points)
5. Quelles sont les affirmations concernant le modèle de conception d'Observer que vous approuvez ? (2 points)

   ☐ Lorsque l'interface d'un sujet change, cela a également des répercussions sur l'observateur.

   ☐ Avant qu'un observateur ne soit informé des changements d'état d'un sujet, il doit s'enregistrer auprès du sujet.

   ☐ Les sujets actualisent leur état dès que des observateurs s'enregistrent auprès des sujets.

   ☐ Un observateur appelle sporadiquement la méthode `notify()` duSujet fait appel à la méthode de l'objet, pour d'être informé des changements d'état du sujet.

   ☐ L'observateur doit implémenter une interface définie dans une classe de base abstraite à partir de laquelle l'observateur est dérivé.