#No: Name Surname/Ad Soyad: o 1. öğr. o 2. öğr.

Object Oriented Programming / Resit Exam - *Nesne Yönelimli Programlama / Bütünleme Sınavı*

Q1/S1) A class named myString that contains a dynamic array of characters are used in main(). - *main() içeriği yanda verilen programda içeriğinde dinamik bir karakter dizisi barındıran myString adında bir sınıftan nesneler oluşturulmaktadır* (40p).

```
int main() {
  char cstr[] = "Zafer";
  myString s1(cstr),s2;
  s1.upperCase();
  s2.lowerCase();
  myPrint(s1);    myPrint(s2);
  myString s3 = s2 + s1;
  myPrint(s3);    }
```

a) The definition of the myString class is given below. Add a friend function in the blanks named myPrint(...) to write the string to the screen. - *myString sınıfının tanımı aşağıda verilmiştir. Sınıf tanımında boş bırakılan alana sınıftaki karakter dizisini ekrana yazacak myPrint(...) adında bir arkadaş fonksiyon ekleyiniz* (8p).

```
class myString{
   char *arr;
   int size;
  public:
   myString();
   myString(char *_str);
   void setString(char *_str);
   void upperCase();
   void lowerCase();
   _____________________________________   // prototype, 2p
   myString operator+(myString &_myStr);
};
_______ myPrint(_____________________){          // 2+2p
      _____________________________;          } // 2p
```

b) Fill in the blanks in the constructor and setter functions. The default string value is "Merhaba C ++". - *Varsayılan string değeri "Merhaba C++" olacak şekilde yapıcı fonksiyonlarda ve setter fonksiyonunda boş bırakılan yerleri doldurunuz* (8p).

```
myString::myString(){
  ____________________________          // 2p
  setString(c);                         }
myString::myString(char *_str){ ____________________  /* 2p */  }
void myString::setString(char *_str){
  int i=0;
  while(_________________) _____________; // dizinin boyutunu bul, 1+1p
  size = i+1;          arr = new char[size];
  for(int i=0;i<size;i++) _____________________;          // 2p
  arr[i] = '\0';                                         }
```

c) Fill in the blanks in upperCase() and lowerCase () functions that make characters upper or lower - *Karakterleri büyük yapan upperCase() ve küçük yapan lowerCase() fonksiyonlarında boş bırakılan alanları doldurunuz* (8p).

```
void myString::lowerCase(){
  for(int i=0;i<size;i++)
    if(arr[i] >= 'A' && arr[i] <= 'Z'){
      ____________________________________          // 4p
    }  }
void myString::upperCase(){
  for(int i=0;i<size;i++)
    if(_____________________________){          // 2p
      ____________________________          // 2p
    }  }
```

d) Rewrite the for loop in the lowerCase () function given in item c with the while loop. - *c maddesinde verilen lowerCase() fonksiyonunda yer alan for döngüsünü aynı işi yapacak şekilde while döngüsü ile yeniden yazınız* (6p).

```
void myString::lowerCase(){




                                                    }
```

e) Fill in the blanks in the function written for (+) operator. The function is used to add 2 strings consecutively. Note that the end of the string is a NULL character and is included in the total string size. - *(+) operatörü için yazılan fonksiyonda boş alanları doldurunuz. Fonksiyonun görevi 2 stringi arka arkaya eklemektir. stringlerin sonunda NULL karakteri olduğuna ve bunun string toplam boyutuna dahil olduğuna dikkat ediniz* (6p).

```
myString myString::operator+(myString &_myStr){
  int i, nsize = ____________________________          // 2p
  char *temp = new char[nsize];
  for(i=0;this->arr[i]!=0;i++)
    temp[i] = ____________________________          // 2p
  for(int j=0;j<_myStr.size;j++,i++)
    temp[i] = ____________________________          // 2p
  temp[i] = 0; // NULL
  return myString(temp);                          }
```

f) Write the screen output when the main func. is executed. - *main içeriği çalıştırıldığında oluşan ekran çıktısını yazınız* (4p).

Q2/S2) In the program that the main () content is given below, you are asked to define a pure virtual class called Vehicle and classes derived from this class called Diesel and Electric. The properties of these classes are given below. - *Son yıllarda elektrikli araçların yaygınlaşması bu araçların hangi usulle vergilendirileceği konusunu gündeme getirmektedir. Aşağıda main() içeriği verilen programda sizden Vehicle adında bir saf sanal sınıf ve bu sınıftan türetilen Diesel ve Electric sınıfları tanımlamanız istenmektedir. Bu sınıfların özellikleri aşağıda verilmektedir* (30p).

- Vehicle class has a string variable named "name". - *Vehicle sınıfı miras yolu ile aktarılacak string türünde "name" verisine sahiptir.*
- Vehicle class has a ctor that takes a parameter which default value is "Vehicle A", a pure virtual function named getTax(..) which returns a double, a getter function named getName() which returns string. - *Vehicle sınıfı varsayılan değeri "Vehicle A" olan string türünden bir parametre alan yapıcı bir metoda, geridönüş değeri double olan getTax(void) isminde saf sanal bir methoda ve string değerini geri döndüren getName() isimli bir getter mothoda sahiptir.*
- Diesel class has a double variable named "motorCC". - *Diesel sınıfı double türünde motorCC adında bir veriye sahiptir.*
- Diesel class has a ctor that takes 2 params which default values are "TOFAŞ SLX" ve 1600, the "name" variable should be sent to super class as parameter. - *Diesel sınıfı varsayılan değeri "TOFAŞ SLX" ve 1600 olan 2 parametre alan ve "name" verisini üst sınıfın yapıcı metoduna parametre gönderme yöntemi ile setleyen bir yapıcı metoda sahiptir.*
- Diesel class has getTax(..) method that calculates the tax according to equation **(500 + (motorCC - 1300)*20)**. *Diesel sınıfı (500 + (motorCC - 1300)*20) eşitliği ile vergi hesaplayan getTax() metoduna sahiptir.*
- Electric class has a double variable named "range". - *Electric sınıfı double türünde "range" adında bir veriye sahiptir.*
- Electric class has a ctor that takes 2 params which default values are "TOGG A" ve 500, the "name" variable should be sent to super class as parameter. - *Electric sınıfı varsayılan değeri "TOGG A" ve 500 olan 2 parametre alan ve name verisini üst sınıfın yapıcı metoduna parametre gönderme yöntemi ile setleyen bir yapıcı metoda sahiptir.*
- Diesel class has getTax(..) method that calculates the tax according to equation **(300 + (range-200)*10)**. - Electric sınıfı (300 + (range-200)*10) eşitliği ile vergi hesaplayan getTax() metoduna sahiptir.

```
int main(){
  Diesel d1,d2("Ford Ka",1300);
  Electric e1,e2("Tesla Model 3",400);
  Vehicle *vptr[5];
  vptr[0] = &d1;  vptr[1] = &d2;
  vptr[2] = &e1;  vptr[3] = &e2;
  findMaximumTax(vptr,4);
  return 0;
}
```

a) Write the definitions of the Vehicle, Diesel and Electric classes - *Vehicle, Diesel ve Electric sınıflarının tanımlarını yazınız.* (1x11p).

```
class Vehicle{
 protected:
   _____________________
 public:
   ____________________________
   ____________________________
   ____________________________                    };
```

```
class Diesel : _____{
  _____
 public:
  _____________________________________________________
  _____________________________                    };
class Electric : _____________________{
   double range;
 public:
  _____________________________________________
  _____________________________                    };
```

b) Write constructor methods of Vehicle, Diesel and Electric classes - *Vehicle, Diesel ve Electric sınıflarının yapıcı metodlarını yazınız* (6p).

```
Vehicle::Vehicle(string _name){ _____________________ }   // 2p
Diesel::Diesel(string _name, double _motorCC):Vehicle(_name){
 _____________________________                    }        // 2p
Electric::Electric(string _name,double _range):Vehicle(_name){
 _____________________________                    }        // 2p
```

c) Write the getName (..) method of the Vehicle class and getTax (..) method of the Diesel and Electric class. - *Vehicle sınıfının getName(..) ve Diesel ve Electric sınıfının getTax(..) metodlarını yazınız* (5p).

```
string Vehicle::getName(){   _________________   }          //1p
double Diesel::getTax(){   ____________________________   }   // 2p
double Electric::getTax(){  ____________________________   }  // 2p
```

d) Fill in the blanks in the findMaximumTax (..) method that displays the name and tax of the vehicle which has the maximum tax. Function takes Vehicle pointer arrays and array size as parameters - *Maksimum vergisi olan aracın adını ve vergi miktarını ekrana yazan findMaximumTax(..) metodunda boş alanları doldurunuz. Fonksiyon parametre olarak Vehicle türünden pointer dizisi ve dizinin boyutunu almaktadır* (4p).

```
void findMaximumTax(_____________________________){    // 2p
 double max = -10;
 int maxi = 0;
 for(int i=0;i<n;i++)
  if(ptr[i]->getTax() > max){
   _____________________________________              // 2p
   maxi = i;
  }
 cout<<"Maximum Tax: ";
 cout<<ptr[maxi]->getName()<<" - "<<ptr[maxi]->getTax()<<endl;          }
```

e) Write the screen output when the main func. is executed. - *main içeriği çalıştırıldığında oluşan ekran çıktısını yazınız* (4p).

Q3/S3) A class structure for fifo and stack data types are given below. The stack data type accepts int data and works in last in first out fashion. The fifo data type implements a circular buffer and operates in a first in first out fashion. *Fifo ve stack için sınıf yapıları aşağıda verilmiştir. Stack sınıfı veriyi son giren ilk çıkar şeklinde işler. Fıfo sınıfı ise veriyi ilk giren ilk çıkar şeklinde işler ve dairesel bir fifo yapısını temsil eder. Burada fifo sınıfı son okunan eleman için readIndex ve son yazılan eleman için writeIndex verilerini tutar. GetCurrentDepth fonksiyonu fifoda bulunan eleman sayısını geri çevirir. Bu fonksiyon verileri kopyalama sırasında kullanılabilir. Özellikle bu soruda gerçeklenmesi gereken operatör fonksiyonları için gereklidir* (30p).

yan sütundan devam ediniz

Exam Rules - *Sınav Kuralları*

1. It is compulsory to follow the rules of the examination announced on the web site - *Bölüm sayfasında ilan edilen sınav kurallarına uymak zorunludur.*
2. Questions are associated with program outputs 2 and 3. - *Sorular 2 ve 3 numaralı program çıktıları ile ilişkilidir.*
3. Ensure that the function prototypes are correct and the written code is executable in all questions . - *Tüm sorularda fonksiyon prototiplerinin doğru olmasına ve yazılan kodun çalıştırılabilir olmasına dikkat ediniz.*

GOOD LUCK - *BAŞARILAR*

```
class bufferType   {
   public:
      bufferType();
      virtual ~bufferType();
      virtual int readItem() = 0;
      virtual int writeItem(int) = 0;
   protected:
      int *buffer;
      int bufSize;
      int writeIndex;
      virtual bool isBufferEmpty()= 0;
      virtual bool isBufferFull() = 0;
};
class stack : public bufferType   {
   public:
      stack();
      stack(long long);
      virtual ~stack();
      virtual int readItem();
      virtual int writeItem(int);
      const stack& operator=(const
stack& stackObj);
   protected:
      virtual bool isBufferEmpty();
      virtual bool isBufferFull();
   private:
};
```

```
class fifo : public bufferType  {
   public:
      fifo();
      fifo(long long);
      virtual ~fifo();
      virtual int readItem();
      virtual int writeItem(int);
      fifo operator+(const fifo&
fifoObj) const;
   protected:
      virtual bool isBufferEmpty();
      virtual bool isBufferFull();
      int getCurrentDepth();
   private:
      int readIndex;
};
```

a) Please create an assignment (=) operator function for **stack class** that copies the content of the right hand side object to left hand side object. If the **left hand side** object has a buffer space that is less than the **right hand side** object the buffer space should be resized. *Atama operatör fonksiyonunu stack sınıfı için gerçekleyiniz. Burada operatörün sağ tarafındaki nesneye ait veriler sol taraftaki nesneye kopyalanacaktir. Eğer sol taraftaki nesnenin tamponu yeterince büyük değilse tampon genişletilmelidir.*

```
const stack& stack::operator=(const stack& stackObj){
  if(_____________________________) { //iki nesne aynı olmamalı
    if(_____________________________){//sol taraftaki nesne daha büyük
tampona sahip, normal kopyalama
      _____________________________
      for(int _______; _______< _____________; _______)
            _____________________________;
    } else {        //Daha büyük bir tampon gerekli, büyük tampon oluştur
      _____________________;
      _____________________________________;
      _____________________________;
      _____________________________;
      for(int _____________; _______ < _____________; _________) //Şimdi kopyala
            _____________________________________;
    }
  }
  return _____________________________________;                }
```

b) Please create an addition (+) operator function for **fifo class** that adds the content of the **first** object to **second** object by creating a third object that can accommodate both objects contents. The resulting object should have a buffer space that is as large as the input objects buffer spaces added together. The contents of the input objects should be copied to the result object starting from the first buffer location. *Toplama operatörünü fifo sınıfı için gerçekleyiniz. Burada iki nesne toplanarak bu nesnelerin toplam boyutu kadar veriyi tutabilecek üçüncü bir object oluşturulmalıdır. Yeni obje oluşturulduktan sonra önceki objelerdeki veriler uygun şekilde yeni objenin ilk veri saklama noktasından başlayarak yeni nesneye kopyalanmalıdır.*

```
fifo fifo::operator+(fifo& fifoObj){
   fifo nFifo(_____________________________);
   int fIdx;                                        //ilk for
   for( fIdx = 0; _______ < _____________; fIdx ++)
          _____________ = _____________________________;
   nFifo.bufSize = _____________________________;      //ikinci for
   for(fIdx= nFifo.bufSize;fIndex < (_____________________); fIdx++)
        nFifo.buffer[fIdx] =_____________________________________;
   nFifo.writeIndex = _____________________________________;
   nFifo.readIndex = 0;
   return nFifo;                                    }
```