Indian Institute of Technology Kharagpur
Mid-Spring Semester Examination 2022-23

Date of Examination: Feb, 2023 **Duration:** 2 Hours
Subject No.: CS20006/CS20202 **Subject:** Software Engineering
Department/Center/School: Computer Science **Credits:** 3 **Full marks:** 50

Name: _____

Roll Number: _____

Instructions

i. **Please write your name and roll number above before attempting any solution**.
ii. **All questions are compulsory.** Be brief and precise. Mysterious or unsupported answers will not receive full marks.
iii. Write your answers in the space provided in this question paper booklet.
iv. Answer to each question should be strictly within the space provided for that question. If the space provided for the answer is not sufficient, the rest of the answer may be written in the Extra Space provided after all questions. This should be clearly indicated in the space provided, failing which, the extra answer may not be checked.
v. Use of electronic calculators only is permitted. No extra resources viz. graph papers, log-tables, trigonometric tables would be required.

Question:	1	2	3	4	Total
Points:	15	10	10	15	50
Score:					

1. (a) (1 point) Consider the following code segment.

```
1 #include <iostream>
2 #include <stdlib.h>
3 using namespace std;
4
5 int main() {
6     int *p = _____________;
7     cout << *p; free(p)
8     return 0;
9 }
```

Fill in the blank at `line 6` such that the output is `10`. The options are
(a) `(int*)malloc(10*sizeof(int))` (b) `new int` (c) `new int(10)` (d) `new int[10]`

> **Solution:** (c) `new int(10)`

(b) (1 point) Consider the following code segment.

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int a = 3, b = 5;
6     ________ int &c = ++a + ++b;    //LINE-1
7     cout << c;
8     return 0;
9 }
```

Fill in the blank at `line 6` such that the output is `10`. The options are (a) `static` (b) `volatile` (c) `const` (d) `inline`

> **Solution:** (c) `const`
> **Explanation**: The result is assigned to the reference variable `c`. The result is stored in a temporary address which is referred by the variable `c`. Hence, `c` should be constant.

(c) (1 point) Consider the following code segment.

```
1 class employee {
2     // some code
3 };
4
5 const employee e_obj;
```

From among the given options, which are the appropriate type(s) for the `this` pointer corresponds to the object `e_obj`?

(a) `employee * const this`
(b) `employee const * this`
(c) `employee const * const this`
(d) `const employee * const this`

Solution: (c) and (d)

(d) (1 point) Consider the following code segment.

```
1  #include <iostream>
2  using namespace std;
3
4  class Item {
5     int value;
6  public:
7     void setValue(int value);
8     int getValue() const { return value; }
9  };
10 void Item::setValue(int value){
11    value = value;
12 }
13
14 int main(){
15    Item x;
16    x.setValue(2);
17    // Now print out the value...expecting 2
18    cout << x.getValue() << endl;
19    return 0;
20 }
```

Examine the `setValue()` function between line `10` and `12`. Something is wrong with it. Fix it without changing the signature of the function.

Solution: `this->value = value;`

(e) (1 point) Consider the following code segment.

```
#include <iostream>
using namespace std;

class mClass {
    int a, b, c;
    public:
        mClass(int v_ = 0) : c(++v_), b(v_++), a(++v_) { }
        void print()
            { cout << "a = " << a << ", b = " << b << ", c = " << c <<
    endl; }
};

int main() {
    mClass obj(10);
    obj.print();
    return 0;
}
```

What will be the output?

Solution: `a = 11, b = 11, c = 13` as initialization follows the list of declaration.

(f) (1 point) Consider the following code segment.

```
1  #include <iostream>
2  using namespace std;
3
4  int divide(int a, int b){
5      return a/b;
6  }
7
8  int main(){
9      cout << divide(015,2) << endl;
10 }
```

What will be printed in the console?

Solution: 6. (`015` is `15` in octal notation.)

(g) (2 points) Consider the following code segment.

```
1  #include <iostream>
2  using namespace std;
3
4  class employee {
5      int eid; string name;
6      employee(){ }
7      public:
8          employee(int eid_, string name_) : eid(eid_), name(name_){ }
9          void updateEmployee(employee& e){ this = &e; }
10         void print(){ cout << eid << ", " << name << endl; }
11 };
12
13 int main() {
14     employee e1(10, "Bhaskar");
15     employee e2(20, "Piyush");
16     e1.updateEmployee(e2);
17     e1.print();
18     return 0;
19 }
```

What will be the output/error? Briefly justify your answer.

Solution: Compiler error at line `9` as `this` being a constant pointer, any attempt to modify it (at line `9`) results in a compiler error.

(h) (4 points) Consider the following code segment.

```
#include <iostream>
using namespace std;

class point{
    int x, y;
    public:
        point(int x_ = 0, int y_ = 0) : x(x_), y(y_){cout << "ctor ";}
        point(point &p) : x(p.x), y(p.y){cout << "c-ctor ";}
        point& operator=(point p)
            {x = p.x; y = p.y; cout << "c-assign "; return *this;}

};

int main() {
    point p1(10, 20);
    point p2 = p1;
    point *pt;
    point p3;
    p3 = p2;
    return 0;
}
```

What will be the output/error? Briefly justify your answer.

Solution: `ctor c-ctor ctor c-ctor c-assign`
The statement `point p1(10, 20);` invokes the constructor and prints `ctor`. The statement `point p2 = p1;` invokes the copy constructor and prints `c-ctor`. The statement `point *pt;` does not create an object. The statement `point p3;` invokes the constructor and prints `ctor`. The statement `p3 = p2;` calls the copy assignment operator. Since it passes `p2` by value, it invokes copy constructor and prints `c-ctor`. Then executes the body of the copy assignment operator and prints `c-assign`.

(i) (3 points) Consider the following code segment. Fill in the blanks as per the instructions such that it satisfies the given test cases.

- at `LINE-7` with appropriate initialization block to initialize the data members,
- at `LINE-9` with appropriate definition of the destructor to free the memory allocated for the data members,
- at `LINE-11` and `13` with appropriate definition of the functions `getX()` and `getY()`,

```
1  #include <iostream>
2  using namespace std;
3
4  class point {
5      const int *px, *py;
6      public:
7          point(int x, int y) : _____________________________________{ }
8
9          ~point() { ____________________________________________ }
10
11         int getX() { ____________________________________________ }
12
13         int getY() { ____________________________________________ }
14 };
15
16 int main() {
17     int i, j;
18     cin >> i >> j;
19     point p(i, j);
20     cout << "[" << p.getX() << ", " << p.getY() << "]";
21     return 0;
22 }
```

Test Case I	**Test Case II**
`Input: 5 6`	`Input: 10 20`
`Output: [5, 6]`	`Output: [10, 20]`

Solution:
```
LINE-7:  px(new int(x)), py(new int(y))
LINE-8:  delete px; delete py;
LINE-9:  return *px;
LINE-10:  return *py;
```

2. (a) (4 points) Consider the following code segment. Complete two functions `swap` and `reverse`. `swap` performs a swap using pass-by-reference. `reverse` reverses a vector using pass by reference. if `vec` = [2, 0, 1, 5] in that order, after calling `reverse(vec)`, it should contain [5, 1, 0, 2] in that order. Call the `swap` function you wrote inside `reverse` to help you.

```
#include <iostream>
#include <vector>
using namespace std;
// Complete this function to swap using pass-by-reference

void swap(_____________________________________)
{
_____________________________________________________________________________

_____________________________________________________________________________

_____________________________________________________________________________

_____________________________________________________________________________

}

// complete this function to reverse a vector

void reverse(_____________________________________)
{
_____________________________________________________________________________

_____________________________________________________________________________

_____________________________________________________________________________

_____________________________________________________________________________

}
void print_vec(vector<int> V){
   for (int i=0; i<V.size(); i++)
      cout << V[i] << " ";
   cout << endl;
}

int main() {
   vector<int> test{ 2, 0, 1, 5};
   cout << "Before reverse: "; print_vec(test);
   reverse(test);
   cout << "After reverse: "; print_vec(test);
}
```

Solution:

```
// Performing a swap using pass-by-reference
void swap(int& a, int& b){
   int t = a;
   a=b;
   b=t;
}

// complete this function to reverse a vector
void reverse( vector<int>& V ) {
  for (int i=0; i<V.size()/2; i++)
    swap(V[i],V[V.size()-i-1]);
}
```

(b) (6 points) You have seen example of overloading + operator for strings that concatenates two strings. You are going to override the * operator to allow multiplication of a string by an int. Python gives the ability to multiply a string constant by an int indicating that you want multiple occurrences of that string. For example: `("hello" * 3)` evaluates to the string `"hellohellohello"`. we can introduce this behavior for string objects in c++ by overloading the * operator. Overloading the * operator in this way will require two different overloads that deal with the two cases of whether the string comes first or the int comes first. You may assume that the integer passed to the function is not negative. Write your code in the space provided below. The expected output is:
`HelloHelloHello`
`HelloHelloHelloHelloHello`.

```
#include <iostream>
using namespace std;

// Start of your code









// End of your code

int main(){
  string str = "Hello";
  cout << str*3 << endl;
  cout << 5*str<< endl;
}
```

Solution:

```
string operator*(const string& lhs, int rhs){
  string result;
  for(int i=0; i<rhs; i++)
    result+=lhs;
  return result;
}

string operator*(int lhs, const string& rhs){
  return rhs*lhs;
}
```

3. Write the outcomes of compiling and executing the following programs? Some of the programs may suffer from compilation errors. For them, write the line number and the reason for the error. Marks will not be given for simply writing "compilation error". For programs that compile, write the output.

(a) (2 points) What is the outcome for the following program:

```
1  #include <iostream>
2  using namespace std;
3
4  class A {
5     public:
6     const int *vals;
7     A(int *array): vals(array) {}
8  };
9
10 int main() {
11    int ar[] = {1,2,3};
12    A a(ar);
13    *(a.vals+1)=4;
14    cout<< a.vals[1]<<endl;
15 }
```

Solution: compilation error in `*(a.vals+1)=4;`

(b) (2 points) What is the outcome for the following program:

```
1  #include <iostream>
2  using namespace std;
3
4  class A {
5     public:
6     int val;
7     A(const int &input): val(input) {}
8  };
9
10 int main() {
11    int i=2;
12     A a(i);
13    a.val=3;
14    cout<< a.val<<endl;
15 }
```

Solution: Output: 3

(c) (2 points) What is the outcome for the following program:

```
#include <iostream>
using namespace std;

class A;
class B {
   int bval;
   public:
   B(int b=5): bval(b) {}
   double convcomb(const A& a, double lambda=0.5);
};
class A {
   int aval;
   public:
   A(int a=0): aval(a) {}
   friend double B::convcomb(const A &, double);
};
double B::convcomb(const A& a, double lambda) {
   return (lambda*bval + (1-lambda)*a.aval);
}

int main() {
   A a;
   B b;
   cout<<b.convcomb(a,0.5)<<endl;
}
```

Solution: Output: 2.5

(d) (2 points) What is the outcome for the following program:

```
#include <iostream>
using namespace std;

class A;
class B {
   int bval;
   public:
   B(int b=5): bval(b) {}
   double convcomb(const A& a, double lambda=0.5);
};
class A {
   int aval;
   public:
   A(int a=0): aval(a) {}
   friend double B::convcomb(const A &);
};
double B::convcomb(const A& a, double lambda) {
   return (lambda*bval + (1-lambda)*a.aval);
```

```
19 }
20
21 int main() {
22    A a;
23    B b;
24    cout<<b.convcomb(a,0.5)<<endl;
25 }
```

Solution: Compilation error: `a.aval` not accessible in `return (lambda*bval + (1-lambda)*a.aval);`

(e) (2 points) What is the outcome for the following program:

```
1  #include <iostream>
2  using namespace std;
3
4  class A;
5  class B {
6      int bval;
7      public:
8      B(int b=5): bval(b) {}
9      double convcomb(const A& a, double lambda=0.5);
10 };
11 class A {
12     int aval;
13     public:
14     A(int a=0): aval(a) {}
15     friend class B;
16 };
17 double B::convcomb(const A& a, double lambda) {
18     return (lambda*bval + (1-lambda)*a.aval);
19 }
20
21 int main() {
22     A a(23);
23     B b(35);
24     cout<<b.convcomb(a,1)<<endl;
25 }
```

Solution: Output : 35

4. Consider a class `MyIter`, which implements an iterator over a vector, such that there can either be many read-only instances of the iterator or at most one readable-writable instance of the iterator. The read-only instances are created as constant instances of the class through `getreadable` static function and read-write instance are created through `getwritable` static

function. The constructor is private to prevent other ways of getting an instance, but the destructor public.

Fill in the functions in the code below so that the following main function produces the indicated output. You are **not** allowed to create extra member variables.

```
#include <iostream>
#include <vector>
using namespace std;

class MyIter {
   unsigned int start,end;
   mutable unsigned int current;
   vector<int> & vec;
   static int numreadableinst; // number of instances of readable objects
   static bool haswritableinst; // is there a writable object

   _________________Constructor____________________

   public:
   _________________Destructor_____________________

   _________________Function getreadble_____________

   _________________Function getwritable_____________

   _____________Member function definitions__________
};

//Initial state
int MyIter::numreadableinst = 0;
bool MyIter::haswritableinst = false;

int main() {
   vector <int> v1(10);
   MyIter * w = MyIter::getwritable(v1);
   int i=1;
   for(w->begin();!w->isend();w->next()) w->set(i++);
   MyIter * w1 = MyIter::getwritable(v1);
   const MyIter * r2 = MyIter::getreadable(v1);
   cout<<"w: "<<w<<" w1: "<<w1<<" r2: "<<r2<<endl;
   delete w;
   const MyIter * r = MyIter::getreadable(v1);
   for(r->begin();!r->isend();r->next()) cout<<r->get()<<endl;
   const MyIter * r3 = MyIter::getreadable(v1);
   cout<<"r: "<<r<<" r3: "<<r3<<endl;
   delete r;
   // MyIter * r = MyIter::getreadable(v1) // Compilation error
}
```

(a) (2 points) Write the constructor.

Solution:

```
MyIter(vector<int> & v, bool writeable)
     : vec(v), start(0), end(v.size()) {
    if(writeable) haswritableinst=true;
    else numreadableinst++;
}
```

(b) (2 points) Write the destructor.

Solution:

```
// Destructor removes instance of readable objects
~MyIter() {
    if(numreadableinst>0) numreadableinst--;
    else haswritableinst=false;
}
```

(c) (3 points) Write the function definition for function `getreadable`.

Solution:

```
// return pointer to a constant MyIter object if no other
//writable MyIter objects exist, NULL otherwise
const static MyIter * getreadable(vector<int> & v) {
    if(haswritableinst) return NULL;
    const MyIter *retval = new MyIter(v,false);
    return retval;
}
```

(d) (3 points) Write the function definition for function `getwritable`.

Solution:

```
// returns pointer to writable MyIter object if no readable
// or writable iterators exist, NULL otherwise
static MyIter * getwritable(vector<int> & v) {
    if(haswritableinst || (numreadableinst>0)) return NULL;
    return new MyIter(v,true);
}
```

(e) (5 points) Write the definitions for other member functions:
`begin`:

Solution:

```
void begin() const {current=start;}
```

`isend`:

Solution:

```
bool isend() const {return current==end;}
```

`next`:

Solution:

```
void next() const {current++;}
```

`set`:

Solution:

```
void set(int a) {vec[current]=a;}
```

`get`:

Solution:

```
int get() const {return vec[current];}
```

Extra Sheets