The University of Nottingham

SCHOOL OF COMPUTER SCIENCE

A LEVEL 2 MODULE, AUTUMN SEMESTER 2008–2009

C/C++ for Java Programmers

Time allowed TWO hours

Candidates may complete the front cover of their answer book and sign their desk card but must NOT write anything else until the start of the examination period is announced

Answer Question ONE and TWO others

Marks available for sections of questions are shown in brackets in the right-hand margin

Only silent, self-contained calculators with a single-line display are permitted in this examination.

Dictionaries are not allowed with one exception. Those whose first language is not English may use a standard translation dictionary to translate between that language and English provided that neither language is the subject of this examination. Subject specific translation dictionaries are not permitted.

No electronic devices capable of storing and retrieving text, including electronic dictionaries, may be used.

DO NOT turn examination paper over until instructed to do so

1 (Compulsory Question)

(*a*) Which of the following statements is/are always true for the following line of code:

```
char* sptr[12];
```

(i) An array of 12 elements is created

(ii) The elements of the array are of type char

(iii) Storage is allocated for a C-style string of length up to 11 characters.

(iv) It is important to set the last element of the array to 0

(v) None of the above (4)

(*b*) It is required to call the function `printf()` if, and only if, the integer variables `x` and `y` are both non-zero. Which (one or more) of the following would have the desired result?

(i) `if ( x & y ) { printf( "i" ); }`

(ii) `if ( x && y ) { printf( "ii" ); }`

(iii) `if ( x | y ) { printf( "iii" ); }`

(iv) `if ( x || y ) { printf( "iv" ); }`

(v) `if ( !(!x || !y) ) { printf( "v" ); }` (4)

(*c*) You are given three files called `main.c`, `list.c` and `list.h`.

The files `main.c` and `list.c` include global functions.

The file `list.h` is a header file which contains the declarations for the functions in `list.c`. You keep the three files in the same directory.

Which of the following, when included near the top of the `main.c` file, would allow the functions which are defined in `list.c` to be used within any functions which are defined in `main.c`?

(i) `import list.*;`

(ii) `import <list.h>`

(iii) `import "list.h"`

(iv) `#include <list.h>`

(v) `#include "list.h"` (3)

(*d*) The following program is intended to demonstrate the usage of the `atoi()` function. It expects one command line parameter and should display the integer value of the parameter. There are two problems with the code, each of which will generate a runtime error under certain circumstances. Briefly describe the problems and the way in which the code should be corrected.

```
int main( int argc, char* argv[] )
{
        int arg1 = atoi(argv[1]);
        if ( argc < 2 )
        {
                printf( "Usage: %s <number>\n", argv[0] );
                return 1;
        }
        printf( "Argument 1 (%s) has value %c\n", argv[1], arg1 );
        return 0;
}
```

(4)

(*e*) What is the output from the following C++ code:

```
int foo( int& a, int b )
{
        static int v = 4;
        a = ++v;
        return b;
}

int main()
{
        int i=1, j=2, k=3;
        i = foo( j, k );
        printf("%d %d %d\n", i, j, k );
        j = foo( i, k );
        printf("%d %d %d\n", i, j, k );
        k = foo( k, j );
        printf("%d %d %d\n", i, j, k );
}
```

(6)

(*f*) Name four methods which may be created implictly by a C++ compiler for a C++ class if they are needed. (4)

2 This question is concerned with `struct`s and linked lists.

(*a*) What is the difference between a `class` and a `struct` in C++? (2)

(*b*) Your employers tell you that they need a system to store sales information for products. Each sale includes a unique sale id number, a product id number, a sales person id number, a transaction quantity and a price.

You are told to store this information in a doubly-linked list.

Write a C `struct` which would be suitable for storing the information for a single sale in a doubly-linked list. (3)

(*c*) You are told to write a function to add a new entry to the end of the linked list. Your function should return an integer value of 1 for success and 0 if there was an error. You are provided with the following function to obtain a unique id number for the transaction:

```
long GenerateUniqueSalesId();
```

The other necessary information should be passed in as parameters to your function. Provide an implementation of the function that will compile on a standard C compiler (not C++ compiler). Your function must both create a list entry and add it to the end of the list. You should use the `struct` which you declared in part (b) to store your list entry.

You should also include a definition for any global variables which you need in order to maintain your list.

Note that the order of the items in the list is irrelevant. You do not need to sort the list, only to add to the end of it. (7)

(*d*) Write a function which takes a single parameter specifying the sale id, finds the item with the specified sale id, removes it from the list and releases any resources which it was using. It should return an integer value of `1` if the item was found and removed or `0` if the item was not found in the list. You should assume that the `struct` which you defined in part (b) is used for the list entries and that they were created using the function which you wrote in part (c). (10)

(*e*) Briefly describe the changes that you would make to the linked list code if you were writing it in C++ instead of C. (Assume that you do not have any access to class libraries so cannot use the standard library classes for linked lists.) (3)

3 This question is concerned with macros, constants and templates.

(*a*) Which (one or more) of the following is the correct syntax for declaring a global constant called `MaxVal` with the value 12.

(i) `const char MaxVal 12;`

(ii) `char const MaxVal 12;`

(iii) `const char MaxVal = 12;`

(iv) `char const MaxVal = 12;`

(v) `const char MaxVal = 12`

(vi) `char const MaxVal = 12` (2)

(*b*) Which (one or more) of the following is the correct syntax for using `#define` to create a constant called `MAXVAL` with the value 5.

(i) `#define MAXVAL 5;`

(ii) `#define MAXVAL 5`

(iii) `#define MAXVAL = 5;`

(iv) `#define MAXVAL = 5`

(v) `#define (MAXVAL) = 5;`

(vi) `#define (MAXVAL) = 5` (2)

(*c*) What is the text that is usually added to header files to prevent the code in the header file being included multiple times (and which thereby avoids cyclic inclusion). Whereabouts in the file is this text placed? (The answer to this question must NOT use the `#pragma` preprocessor directive.) (3)

(*d*) Write a function called `min()` which takes two `long` parameters called `val1` and `val2` and returns the value of the lesser of the two values as a `long` , using the ternary operator (`?:`). (i.e. it should return the minimum of the two values.) (4)

(*e*) Provide the code for a macro called `MIN` using `#define` which will perform the same function as your `min()` function in question 3d. (5)

(*f*) Under what circumstances would the `MIN` macro which you provided the code for in question 3e produce unusual results? (4)

(*g*) A templated function is often a better solution than a macro when programming in C++. Provide the code for a templated function `GetMin` to return the minimum of two values, which takes two parameters of the same type and returns a result of the same type. (5)

4 This question is concerned with identifying the problems with and/or correcting existing code. It is also concerned with `new`, `malloc`, and operator overloading.

(*a*) What are the main differences between `new` and `malloc` ? (4)

(*b*) What is meant by the term 'operator overloading'? (3)

(*c*) The following C++ source code will compile but has two problems which will occur only at runtime (and may or may not not crash the program). Identify the cause of the two run-time problems and suggest a solution for each.

```
#include <cstdio>
#include <cstring>

int main()
{
        char* s1 = new char[30];
        char s2[] = "is not";
        const char* s3 = "likes";

        s3 = "allows";
        strcpy( s2, s3 );
        sprintf( s1, "%s %s %s using functions.",
                "C++", s2, "fast code" );
        printf( "String was : %s\n", s1 );

        delete s1;
}
```

(4)

(*d*) The following C program is supposed to allow the comparison of strings. The comparison should put the strings into alphabetical order. For example, the test strings which are used in the `main()` function below should appear in the order:

```
a aa ab b ba bb
```

However, the function does not work as it should.

```
#include <stdio.h>

int compare( const char* a, const char* b )
{
        const char* p = a;
        const char* q = b;
        char diff;

        while ( *p )
        {
                if ( !*q )
                        return -1;
                diff = *p++ - *q++;
                if ( diff != 0 )
                        return diff;
        }
        if ( !*q )
                return 0;
        return 1;
}



int test( const char* a, const char* b )
{
        int comparison = compare( a, b );
        if ( comparison < 0 )
                printf( "%s comes before %s\n", a, b );
        else if ( comparison == 0 )
                printf( "%s is the same as %s\n", a, b );
        else
                printf( "%s comes after %s\n", a, b );
}

int main()
{
        test( "a", "b" );
        test( "bb", "b" );
        test( "ab", "aa" );
        test( "ab", "bb" );
        test( "aa", "aa" );
        test( "a", "ab" );
}
```

(i) What is the output of this test program? (6)

(ii) What is wrong with the function `compare()` and how could you correct it? (3)

(*e*) The following C code will not compile and link to generate an executable. Even if the compilation errors are corrected, it would have a runtime error. Identify the four problems with the code, three of which will prevent it from compiling or linking and the fourth of which will be the cause of the runtime error.

```
#include <stdio.h>

int main()
{
        char contents[20];
        char* p = contents;
        FILE fptr;
        if ( (fptr=fopen("myfile.dat", "w"))=NULL )
        {
                printf( "Cannot open output\n" );
                exit(1);
        }
        strcpy( contents, "Message to write" );
        while ( *p )
                fputc( *p++, fptr );
        fshut(fptr);
        if ( (fptr=fopen("myfile.dat", "r"))==NULL )
        {
                printf( "Cannot open input\n" );
                exit(1);
        }
        while ( (*p = fgetc(fptr)) != EOF )
                putchar( *p++ );
        fclose(fptr);
        return 0;
}
```

(5)

5 This question is concerned with C++ casts, inheritance and exception handling.

(*a*) Name the four new types of cast that are introduced in C++ (i.e. casts that are not available in C) and give a simple example of when each would be used. (8)

(*b*) What is the output of the following C++ program?

```
#include <cstdio>

class BaseClass
{
public:
        BaseClass() : _val(12) {}
        BaseClass( int val ) : _val(val) {}
        void print() { printf( "Base : %d\n", _val ); }
        void setval( int newVal ) { _val = newVal; }
protected:
        int _val;
};

class DerivedClass : public BaseClass
{
public:
        DerivedClass( int val=6 ) { _val = val; }
        void print() { printf( "Derived : %d\n", _val ); }
};

int main()
{
        BaseClass a(1);
        DerivedClass b;
        BaseClass c = a;
        BaseClass& d = b;
        BaseClass* e = &a;
        DerivedClass* f = &b;
        BaseClass* g = f;

        a.setval(3);
        d.setval(4);

        a.print();
        b.print();
        c.print();
        d.print();
        e->print();
        f->print();
        g->print();

        return 0;
}
```

(7)

(*c*) What is the output of the following C++ code:

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

class Base
{
public:
        virtual void show() { cout << "Green "; }
};

class Sub1 : public Base
{
public:
        void show() { cout << "Blue "; }
};

class Sub2 : public Base
{
public:
        void show()  { cout << "Indigo "; }
};

class Sub3 : public Base
{
public:
        void show() { cout << "Violet "; }
};

int main()
{
        Base* pBase = new Base();
        Sub1* pSub1 = new Sub1();
        Sub2* pSub2 = new Sub2();
        Sub3* pSub3 = new Sub3();

        for ( int i = 0 ; i <= 4 ; i++ )
        {
                try
                {
                        switch( i )
                        {
                        case 0: throw 1;
                        case 1: throw pBase;
                        case 2: throw pSub1;
                        case 3: throw pSub2;
                        case 4: throw pSub3;
                        }
                }
                catch (int param) { cout << "Red "; }
                catch (Sub1* sub1) { cout << "Orange "; }
                catch (Base* base) { base->show(); }
                catch (Sub2* sub2) { cout << "Yellow "; }
                catch (...) { cout << "Unknown "; }
        }
        cout << endl;

        delete pSub3;
        delete pSub2;
        delete pSub1;
        delete pBase;

        return 0;
}
```

(10)