b8

cs FILE COPY

CMPUT 201 Winter Term

Final Exam – Tuesday April 17, 2001

Student ID:

Instructions

- Write your student id number in the box above. Do so **Now**.
- This is an open book exam. Time allowed: **120 minutes**.
- This exam counts 35% towards your final grade in this course. The points for each question are indicated in the square bracket by the question number.
- The exam questions have been broken into 3 groups to help you focus your thoughts. Some questions do not have obvious answers. Think about each question group for a while before answering.

Carefully note how many marks there are for each question and allocate your time appropriately.

- Do the parts of the questions you find easiest, **FIRST**.
- There are 3 sections and 6 sheets (12 sides, including this cover page) in the exam booklet. You are responsible for checking that your exam is complete. Do so **Now**.

	Section	Mark	Out of
	Basic C		30
	Labs and Methodology		26
	Fundamental C++		44
	Total		100

Examination prepared by Professor Tony Marsland, April 2001.

Section 1 [30 Marks in total] Questions related primarily to basic C.

Question 1.1 [5 Marks]: Identify three important problems found in the following program fragment. In each case, be explicit and state the exact nature of the problem.

```
int j = 5;
int* b;
int* q = ( int* ) calloc (10, sizeof*b);

if ( q == NULL ) {
      fprintf (stderr, "Null pointer error.\n");
}
b = q;
q = q + 5;
while ( j >= -5 ) {
      *(q+j) = j + 5;
      j--;
};

fprintf (stdout, "\nReclaim the memory\n");
free(b);
free(q);
```

Question 1.2 [4 marks]:
(a) In C, how can you tell if a pointer like `int* ptr` points to a single int, or to an array of integers?

(b) What are the main differences between x and y in the following two statements

```
      char* x = (char*) malloc (15*sizeof(char));
      char y[15];
```

Consider at least whether the following uses of x and y equally valid:
*x = 0; *y = 0; x++; y++; Give a reason for each answer.
Also, what are the values of sizeof(x) and sizeof(y)?

Question 1.3 [Marks 5]: Given

```
typedef struct student {
      char* name;
      int grade;
} Student;
Student group [100];
```

Write a comparison function called `cstudent` that can be used with `qsort` to sort the `group` array into alphabetic order, based on the name field in each array element. Provide an example of how qsort would use your function. Do not write an entire program, just show a correct invocation of `qsort()`. Show proper casting where appropriate. Prototype is:

```
void qsort (void* , size_t , size_t , int (*fun) (void*, void*) ).
```

Question 1.4 [8 marks]: What does the following procedure do?
Rewrite the procedure so that ALL occurrences of the * and ++ operators are removed. You may add local variables if you wish, but the resultant procedure must do exactly the same job.

```
void strip( char*  dest, const char*  src )
{
   *dest = *src;
   while ( *(src++) )
      if ( *src != *dest )
         *(++dest) = *src;
}
```

Question 1.5 [8 marks]. Using only the three variables `str`, `ptr` and `i`, as specified below, write a code segment which deblanks the given string `str`. That is, when you are done `str` is a null-terminated string that contains no blanks, while retaining the original words. What are the before and after lengths of the string `str`, as given by `strlen()`?

```
char str[100] = "Here we have a general string of characters to deblank";
char* ptr;
int i;
```

Section 2 [26 Marks in total]: Work mainly related to Labs and debugging.

Question 2.1 [2 marks]: C++ allows some sophisticated mechanisms for Data Hiding. Explain in your own words what Data Hiding is, and how it can help us design robust and maintainable programs.

Question 2.2 [2 marks]: In C++, What is meant by the term "Copy Constructor", and why do we frequently need one?

Question 2.3 [4 marks]: Does the following code fragment work as expected? Identify any undesirable properties it may have and suggest improvements.

```
int* adder(int a, int b) {
    int* rp = (int*) malloc(sizeof(int));

    *rp = a + b;
    return rp;
}

printf("The answer is %d\n", *adder(-5, +5) );
```

Question 2.4 [4 Marks]: The following function is syntactically correct, but does not work as intended. Identify and rank the three most important shortcomings, stating the problem in each case.

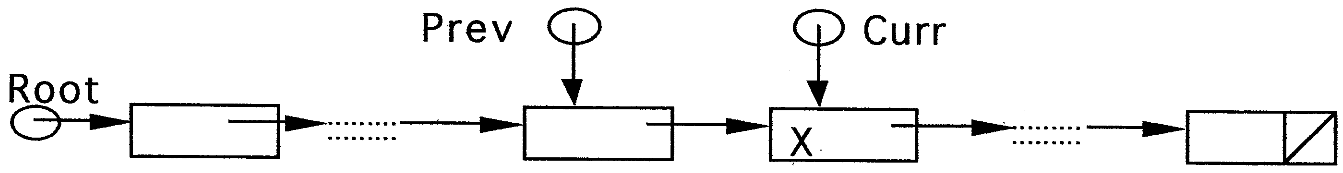
```
char* getline (void) {
        char buffer[80];
        int i, ch;

        i = 0;
        while ( (ch = getchar()) != '\n' && ch != EOF ) {
                buffer[++i]  = ch;,
                ch = getchar();
        }
        buffer[i]  = 0;
        return (&buffer[0]);
}
```

Question 2.5 [4 marks]: Write a simple makefile so that the default target correctly builds an executable `simulator` from the source files `main.cc, data.cc` and `8086.cc`. The routines implemented in `8086.cc` and `data.cc` are both used by `main.cc`, and `data.cc` also requires both prototype header files `data.h` and `8086.h` The usual naming convention for header files applies.

Be sure that the dependencies in the makefile are such that when any one file is changed only that file is recompiled, and no unnecessary work is done.

Question 2.6 [10 Marks] Using the prototypes below, complete the function `Extract` that removes from a linked list the first node whose data value is X. `Extract` returns a pointer to the reduced linked list. The third parameter points to the extracted node, or NULL if X is missing

```
typedef struct item* ItemPtr;
struct item {
     float data;
     ItemPtr next;
};
ItemPtr Extract (Itemptr , float , ItemPtr* );
```



Note: The item to be extracted can be the first item, the last, or any item between, so you will have to traverse the list to find X. Hint: use the variables <u>root</u>, <u>prev</u> and <u>curr</u> appropriately in your solution.

```
ItemPtr Extract(ItemPtr root, float X, ItemPtr* Ptr)
{





}
```

Section 3 [44 marks in total]: C++ questions on classes, derivation and templates

Question 3.1 [15 marks]: Given the code below, what is the output?
For full marks the details must be correct, including all symbols and with correct order of data on each line.

```
/*
* This is a simple 2D vector program that calculates dot products,
* vector sums, and vector differences. There are 2 different constructors
* defined for the Vector class that print out a single string
* ("Create" or "Copy") depending on which constructor is needed.
*/
#include <iostream>
class Vector  {          // Simple 2D vector class
  public:
      int first, second;
      Vector (int f=0, int s=0)
            { first = f; second = s; cout << " Create " << second ; }
      Vector (const Vector& v)
            { first = v.first; second = v.second; cout << " Copy " << first ; }
      Vector& operator=(const Vector& v)
            { first = v.first; second = v.second; cout << " Assign " ; }
      diff (Vector, Vector) ;
};
int product (Vector v1, Vector v2)  {            // Dot product
      int result;
      result = v1.first * v2.first;
      result += v1.second * v2.second;
      return result;
}
Vector sum (Vector& v1, Vector& v2)  {           // Vector sum
      Vector result;
      result.first = v1.first + v2.first;
      result.second = v1.second + v2.second;
      return result;
}
int Vector :: diff (Vector v1, Vector v2)  {    // Vector difference
      first = v1.first - v2.first;
      second = v1.second - v2.second;
}
void main (void) {
      Vector v1(2, 1), v2(3, 4), v3;            // Declare some variables

      Cout << endl;

      cout << " Product is " << product (v1, v2) << endl;

      v3.diff (v1, v2);
      cout << " Diff is <" << v3.first << ", " << v3.second << ">" << endl;

      v3 = sum (v1, v2);
      cout << " Sum is <" << v3.first << ", " << v3.second << ">" << endl;
}
```

Question 3.2 [8 marks]: Given the executable statements:

```
    String S1("Hello"); String S2("There"); String S3;
    S3 = S2;                    // copy S2 into S3
    S3.lesser(S1 , S2);         // S3 contains alphabetically lesser, here "Hello"
```

Modify the following String class declaration so that the above code executes properly. With a few words, justify each modification.

```
class String {
    String (const char* s ) {
        len = 1+strlen(s);
        str = new char[len];
        if(!strcpy(str, s)) exit(1);
    }
private:
    int len;
    char* str;
};
```

Question 3.3 [15 marks]. Consider the following C++ program:

```
#include <iostream>
#include <string.h>

class cow {
   friend ostream& operator<< (ostream &, cow &);
      char* breed;
      int id_tag;
      double milk_production;
      static char* default_breed;
      static int default_id;
public:
      cow (char*, int);
      cow ();
      ~cow ();
      void set_production (double);
      double get_production ();
};
char* cow :: default_breed = "Jersey";
int cow :: default_id = 68040;

cow :: cow (char* c, int id) {
      breed = new char[1+strlen(c)];
      strcpy (breed, c);
      id_tag = id;
      cout << "Constructing a cow of type " << c << endl;
}
cow :: cow () {
      breed = new char[1+strlen(default_breed)];
      strcpy (breed, default_breed);
      id_tag = default_id;
      cout << "Constructing a cow of type " << breed << endl;
}
cow :: ~cow() {
      cout << "Destroying a cow of type " << this->breed << endl;
      delete[] breed;
}
void cow :: set_production (double litres_of_milk) {
      milk_production = litres_of_milk;
}
double cow :: get_production () {
      return (milk_production);
}
ostream& operator << (ostream& s, cow& animal) {
      s << "Cow " << animal.id_tag << " is a " << animal.breed
        << " and should produce "
        << animal.milk_production << " litres of milk." << endl;

      return (s);
}
main() {
      cow one;
      cow two ("Charollais", 8086);
      one.set_production (22.5);
      two.set_production (3.40);

      cout << "Presenting the now famous Tucows ..." << endl;
      cout << one << two;
}
```

[a] What is the significance of the static members in the cow class ?

[b] Could an external function directly access the id_tag of a cow object ? Why ?

[c] Why do we need a special **destructor** for cow objects ?

[d] `ostream& operator<< (ostream&, cow&)` is a **friend** of the cow class. What does this mean?

[e] What does this program print (display on the screen)?

Question 3.4 [6 marks]. Define the three terms: Derivation/Inheritance, Template and Polymorphism as used in C++. Which of these features help support code re-use?. Illustrate each point with a simple example.

Derivation and inheritance:

Template:

Polymorphism: