2014 Paper:

Link to paper: http://exams.victoria.ac.nz.helicon.vuw.ac.nz/handle/123456789/6486

Four steps:

Preprocessing:

Macro substitution, Comments are removed, Expansion of Included files.

Flag = -E

Compilation:

Outputs assembler code. Convert C code to assembly code.

Flag = -s

Assembly:

Output Object code. For each C file, make an executable file. Non human readable. Externally referenced files are not accessible yet.

Flag = -c

Linking;

Link all the files altogether. Create final product. Create executable file.

Flag= -o

Automatic vs Static:

Static

This has scope of only the file itself, and no other files can access these. Stored in a data segment (by itself). Functions that create static variables will persist after the function has been removed from the stack. Other functions cannot access these, but they can use it, if it was returned by the creator function.

Like private in java: It is similar that the variables made in the file will only persist in the file itself. When trying to access them from another file, these cannot be accessed.

Declared outside all blocks or declared inside a block using the *static* keyword.

(In computing, a **data segment** (often denoted **.data**) is a portion of an object file or the corresponding virtual address space of a program that contains initialized static variables, that is, global variables and static local variables.)

e.g.

Automatic:

Created within a function block/function. No storage specification. Like a regular variable, deletes after you leave the function.

http://stackoverflow.com/questions/13415321/difference-between-static-auto-global-and-local-variable-in-the-context-of-c-a

In computer programming, an **automatic variable** is a local variable which is allocated and deallocated automatically when program flow enters and leaves the variable's scope.

Iteration vs Recursion

Less memory used in iteration. Recrusion looks nice. Recursion is intuitive.

TAIL RECURSION: http://stackoverflow.com/questions/33923/what-is-tail-recursion

Recursion:

Looks nice
Requires base case
Intuitive.
Uses more of the stack memory (Cache)

Iteration:

Less Memory
Faster.
Can't get a stack overflow.
More Complex. (Sometimes)

Reallocing and Mallocing

Outcomes:

Realloc could return null.
Tmp is used to check if realloc returns null.
Remember the old pointer, so that we can use it in case the realloc fails.
If ptr is used, and realloc returns null: values that ptr had are lost.
Realloc might not find enough contiguous memory to re allocate the values.

Program problems

Problems:

Doesn't iterate over the whole of tomato string. SIZE is 5 but "tomato" is 6 characters long.

STRCPY: It is copying orange into pear, but pear is only 4 chars long, while orange is 6 chars long.

char v1 doesnt have enough space to contain the last null character at the end of tomato.

(*)STRCAT: f1 doesnt have enough space to copy over f2 also. f1 should have enough space for both things. Undefined operation? Depends on the implementation.

Pointer Declaration

1. True (* Difference between function and pointer to a function)

2. True : returns to a memory address to a function and Fp2 is a pointer (*)
3. True : the function returns an int, and r1 is an int. (*)
4. False: r2 is an int, and fp2 is a pointer to a function that returns (*)

All are true, assigning a function is not as strict as pointers.

1. D OK: its a pointer to a pointer (what the function wants.)
   A OK: 2D array = pointer to a pointer to the first element in the int array (*)
   C OK: is an array of 20 pointers to int, and C is a pointer to the first element in the array.
   B NOT OK: isn't because it's a pointer to an array of 10 elements.A and B are invalid and C and D are valid.

Skipped ahead here: Go to Question 3:

Data Structures

```
void tometer(Height *h){
        if(h->scale == imperial){
                FeetAndInches f = (h->reading).feetandinches;
                (h->reading).meters = (float)((f.feet*0.3048) + (f.inches*0.0254));
                h->scale = metric;
        }
}
```

(* Going into a struct and then into a union?)

Queue Question

```
void enqueue(Data d, Queue *q){
        ptrNode n = malloc(sizeof(struct Node));
        n->data = d;
        n->next = NULL; (* Memory allocation?)
        ptrNode p = q->rear;
        p->next = n;
        q->rear = n;
        (q->cnt)++; (*)
}
```

```
Data dequeue(Queue *q){
        ptrNode n = q->front;
        q->front = n->next;
        Data d = n->data;
        free(n);
        (q->cnt)--;
        return d;
}
```

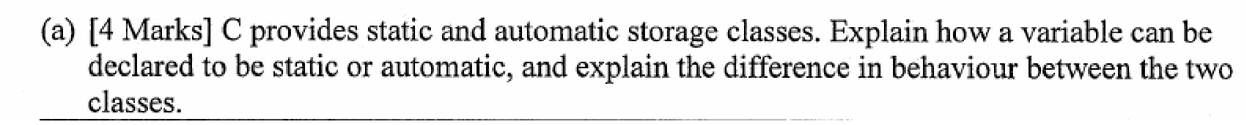
Size of Example: http://pastebin.com/06bwZRc0

Extra Notes on C:

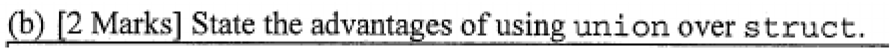
C Programming Study Preparation.

http://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B
http://en.wikipedia.org/wiki/Sizeof
http://en.wikipedia.org/wiki/Bitwise_operations_in_C
http://www.java2s.com/Tutorial/C/CatalogC.htm

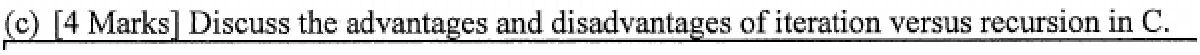
2013 C General Questions.



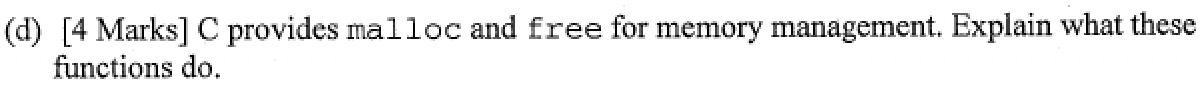
(a) [4 Marks] C provides static and automatic storage classes. Explain how a variable can be declared to be static or automatic, and explain the difference in behaviour between the two classes.



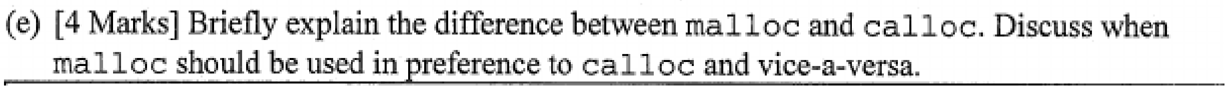
(b) [2 Marks] State the advantages of using `union` over `struct`.



(c) [4 Marks] Discuss the advantages and disadvantages of iteration versus recursion in C.

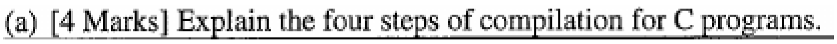


(d) [4 Marks] C provides `malloc` and `free` for memory management. Explain what these functions do.

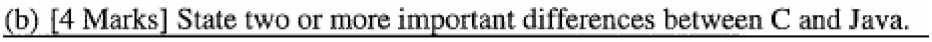


(e) [4 Marks] Briefly explain the difference between `malloc` and `calloc`. Discuss when `malloc` should be used in preference to `calloc` and vice-a-versa.

2012 C General Questions.

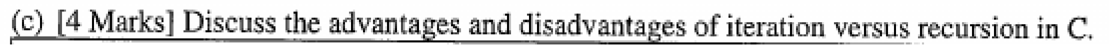


(a) [4 Marks] Explain the four steps of compilation for C programs.

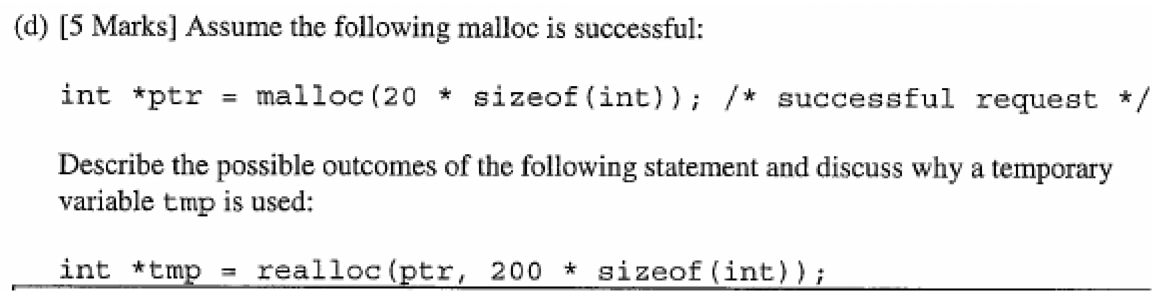


(b) [4 Marks] State two or more important differences between C and Java.

Java uses a syntax similar to C++. C uses for, while and if statements to control the flow of the program. Similarities of c to Java is that java also supports object orientated programming as does c and c++.
C uses stand alone functions. No stand alone in Java. No globals in Java.
The main differences from c to java is that Java does not support a pointer. In c you have to manually free memory, unlike java where memory is cleared by the garbage collection. Java is more portable because it uses byte code and virtual machine. Java also does not support operator overloading.



(c) [4 Marks] Discuss the advantages and disadvantages of iteration versus recursion in C.

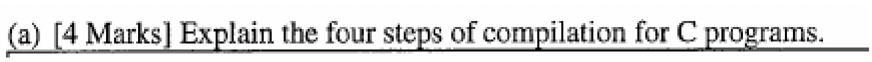


(d) [5 Marks] Assume the following malloc is successful:

```
int *ptr = malloc(20 * sizeof(int)); /* successful request */
```

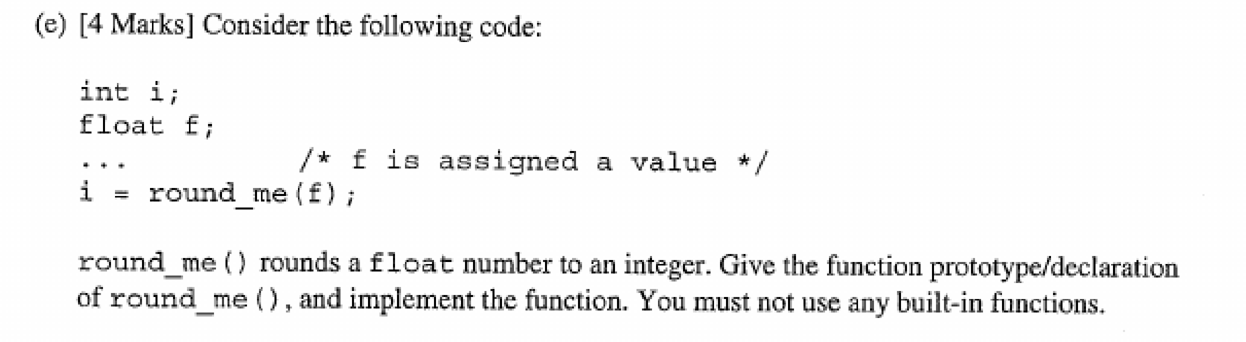
Describe the possible outcomes of the following statement and discuss why a temporary variable tmp is used:

```
int *tmp = realloc(ptr, 200 * sizeof(int));
```



(a) [4 Marks] Explain the four steps of compilation for C programs.

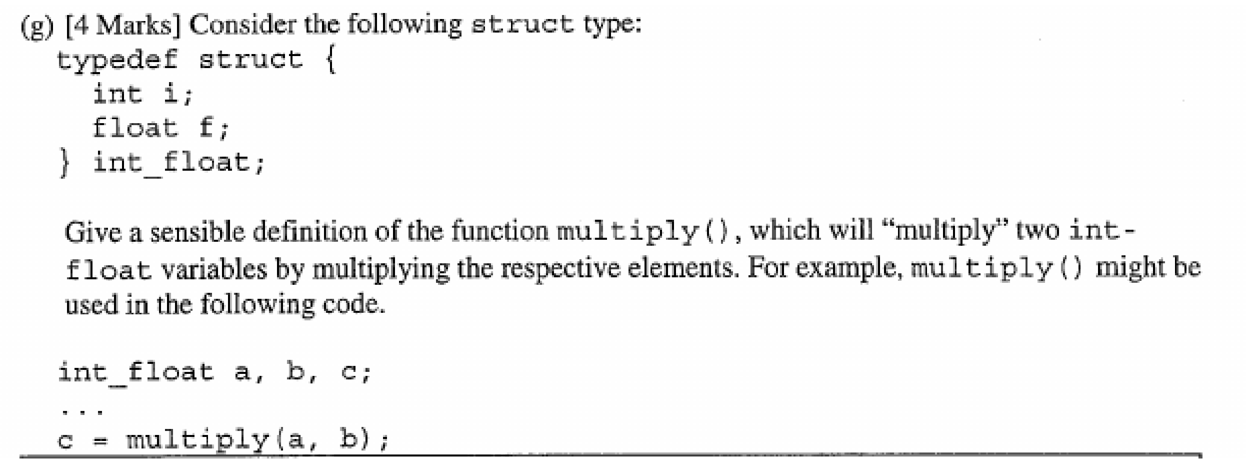
The gcc is responsible for preprocessing, compilation, assembly and linking. Preprocessing is responsible for processing the directives beginning with a '#' character. Preprocessor directives are responsible for text substitution, macro substitution, conditional compilation and inclusion of named files. The purpose of the compilation stage is to make the output assembler code. The purpose of the assembly stage is to make the output object code. Finally the linking stage brings together the assembler code and object code and converts it all into one.



(e) [4 Marks] Consider the following code:

```
int i;
float f;
...          /* f is assigned a value */
i = round_me(f);
```

round_me() rounds a float number to an integer. Give the function prototype/declaration of round_me(), and implement the function. You must not use any built-in functions.

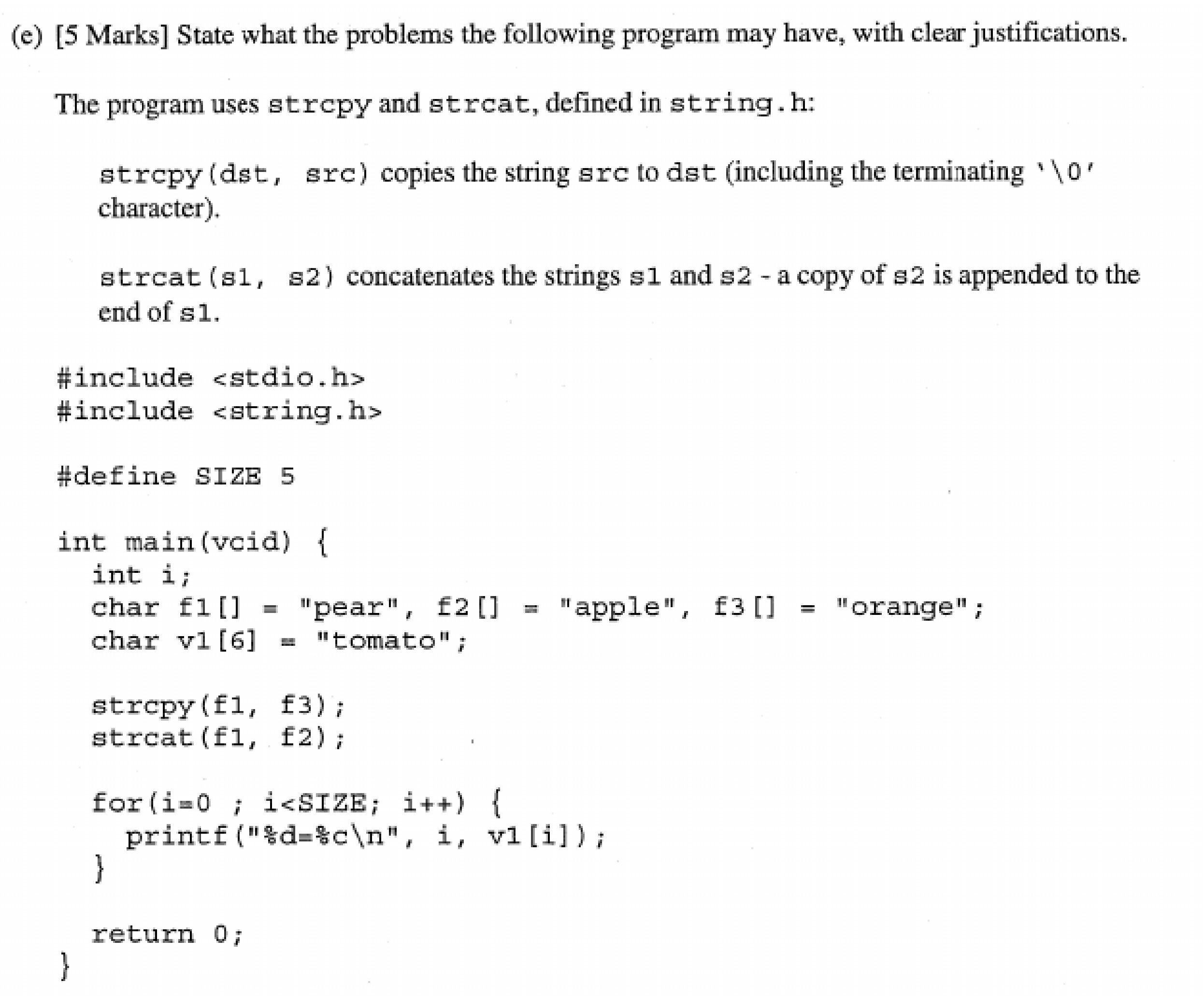


(g) [4 Marks] Consider the following `struct` type:

```
typedef struct {
  int i;
  float f;
} int_float;
```

Give a sensible definition of the function `multiply()`, which will "multiply" two `int-float` variables by multiplying the respective elements. For example, `multiply()` might be used in the following code.

```
int_float a, b, c;
...
c = multiply(a, b);
```



(e) [5 Marks] State what the problems the following program may have, with clear justifications.

The program uses `strcpy` and `strcat`, defined in `string.h`:

`strcpy(dst, src)` copies the string `src` to `dst` (including the terminating `'\0'` character).

`strcat(s1, s2)` concatenates the strings `s1` and `s2` - a copy of `s2` is appended to the end of `s1`.

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

#define SIZE 5

int main(vcid) {
  int i;
  char f1[] = "pear", f2[] = "apple", f3[] = "orange";
  char v1[6] = "tomato";

  strcpy(f1, f3);
  strcat(f1, f2);

  for(i=0 ; i<SIZE; i++) {
    printf("%d=%c\n", i, v1[i]);
  }

  return 0;
}
```

Conversion Character	Displays Argument (Variable's Contents) As
%c	Single character

%d	Signed decimal integer (int)
%e	Signed floating-point value in E notation
%f	Signed floating-point value (float)
%g	Signed value in %e or %f format, whichever is shorter
%i	Signed decimal integer (int)
%o	Unsigned octal (base 8) integer (int)
%s	String of text
%u	Unsigned decimal integer (int)
%x	Unsigned hexadecimal (base 16) integer (int)

%d %i	Decimal signed integer.
%o	Octal integer.
%x %X	Hex integer.
%u	Unsigned integer.
%c	Character.
%s	String. See below.
%f	double
%e %E	double.
%g %G	double.
%p	pointer.
%n	Number of characters written by this printf. No argument expected.
%%	%. No argument expected.

Sequence	Meaning
\a	Beeps the speaker
\b	Backspace (moves the cursor back, no erase)
\f	Form feed (ejects printer page; may clear the screen on some computers)
\n	Newline, like pressing the Enter key
\r	Carriage return (moves the cursor to the beginning of the line)
\t	Tab
\v	Vertical tab (moves the cursor down a line)
\\	The backslash character
\'	The apostrophe
\"	The double-quote character

\?	The question mark
\0	The "null" byte (that's 0, not the letter O)
\Onn	A character value in octal (base 8)
\xnnn	A character value in hexadecimal (base 16)

Notes Below.

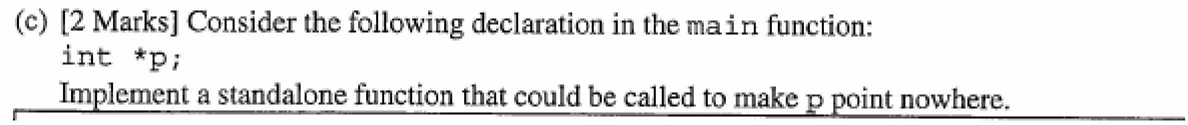
A global variable is available to all functions in your program.

A local variable is available only to the function in which it's created.

1. Global variables are declared outside of any function.
2. Global variables are typically declared right before the main() function.
3. You can also declare a group of global variables at one time:

NEW QUESTIONS ADDED BELOW HERE

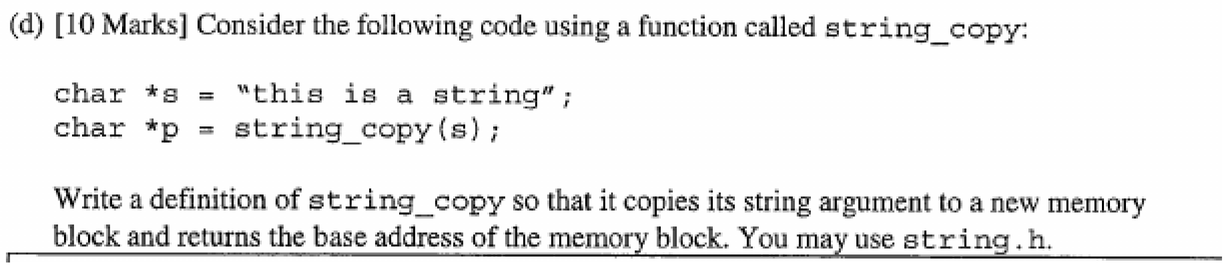
NWEN 241 FINAL EXAM 2012 Question 4



(c) [2 Marks] Consider the following declaration in the `main` function:

```
int *p;
```

Implement a standalone function that could be called to make `p` point nowhere.



(d) [10 Marks] Consider the following code using a function called `string_copy`:

```
char *s = "this is a string";
char *p = string_copy(s);
```

Write a definition of `string_copy` so that it copies its string argument to a new memory block and returns the base address of the memory block. You may use `string.h`.

NWEN 241 FINAL EXAM 2012 Question 5

(a) [4 marks] Consider the following code:

```
typedef struct {
  int i;
  char c;
  int ii;
} int_char_int;

typedef struct {
  int i;
  char c;
  char cc;
  int ii;
} int_char_char_int;
```

Suppose that you work on a 32-bit machine.

What is the value of `sizeof(int_char_int)`?

What is the value of `sizeof(int_char_char_int)`?

Suppose `a` and `p` are defined as follows:

```
int_char_int a;
int_char_int *p = &a;
```

If `p` currently contains the value `m`, what is the value of `p` after the statement `p++;`?

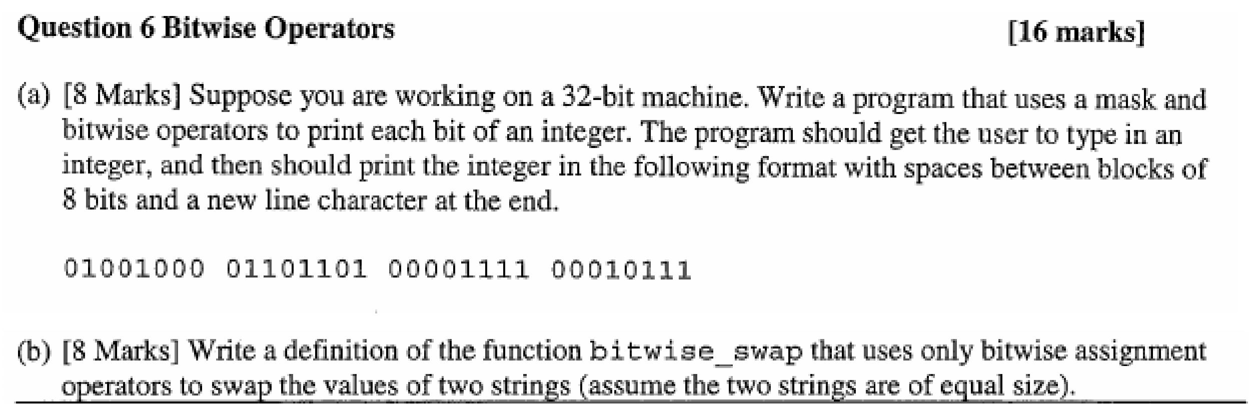
(b) In this question, you need to implement functions that create a singly-linked list using iteration and recursion. You need to use the following type definitions, macro definitions and function prototypes to implement your functions:

```
#define Node_Size sizeof(Node)

typedef struct node
{ char data;
  struct node *next;
} Node;

typedef Node *ptrNode;

ptrNode createlisti(char *); /* iteration */
ptrNode createlistr(char *); /* recursion */
```

i. [8 Marks] Implement the function `createlisti()` using iteration. The function will create a list with one character per node from a string, and return a pointer to the head of the resulting list.

ii. [8 Marks] Implement the function `createlistr()` using recursion. The function will create a list with one character per node from a string, and return a pointer to the head of the resulting list.

NWEN 241 FINAL EXAM 2012 Question 6



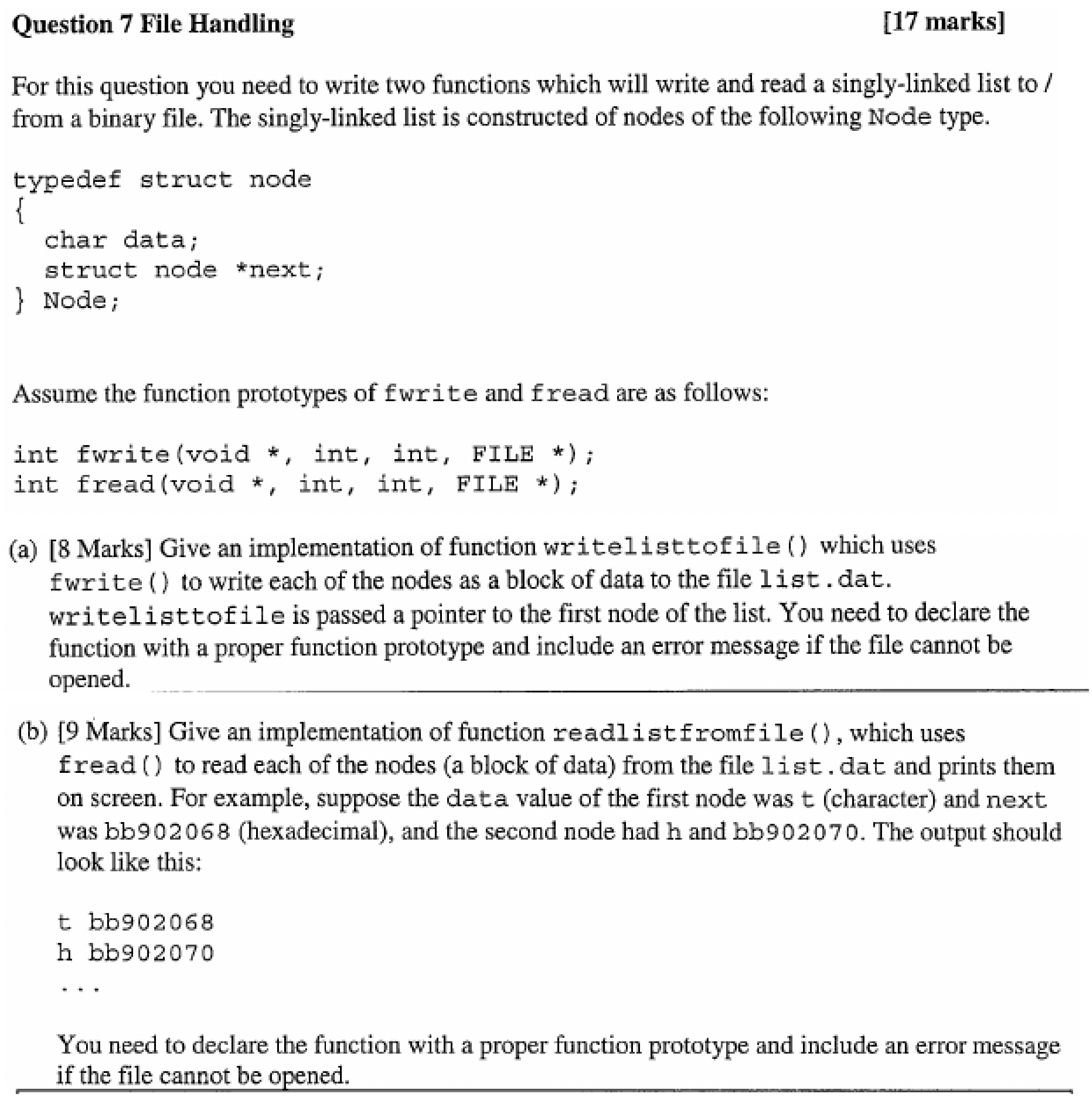
Question 6 Bitwise Operators **[16 marks]**

(a) [8 Marks] Suppose you are working on a 32-bit machine. Write a program that uses a mask and bitwise operators to print each bit of an integer. The program should get the user to type in an integer, and then should print the integer in the following format with spaces between blocks of 8 bits and a new line character at the end.

```
01001000 01101101 00001111 00010111
```

(b) [8 Marks] Write a definition of the function `bitwise_swap` that uses only bitwise assignment operators to swap the values of two strings (assume the two strings are of equal size).

NWEN 241 FINAL EXAM 2012 Question 7



Question 7 File Handling **[17 marks]**

For this question you need to write two functions which will write and read a singly-linked list to / from a binary file. The singly-linked list is constructed of nodes of the following `Node` type.

```
typedef struct node
{
  char data;
  struct node *next;
} Node;
```

Assume the function prototypes of `fwrite` and `fread` are as follows:

```
int fwrite(void *, int, int, FILE *);
int fread(void *, int, int, FILE *);
```

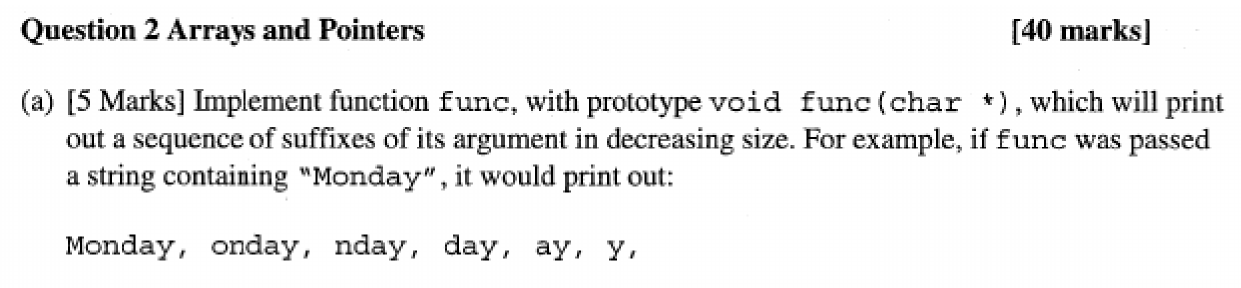
(a) [8 Marks] Give an implementation of function `writelisttofile()` which uses `fwrite()` to write each of the nodes as a block of data to the file `list.dat`. `writelisttofile` is passed a pointer to the first node of the list. You need to declare the function with a proper function prototype and include an error message if the file cannot be opened.

(b) [9 Marks] Give an implementation of function `readlistfromfile()`, which uses `fread()` to read each of the nodes (a block of data) from the file `list.dat` and prints them on screen. For example, suppose the `data` value of the first node was `t` (character) and `next` was `bb902068` (hexadecimal), and the second node had `h` and `bb902070`. The output should look like this:

```
t bb902068
h bb902070
...
```

You need to declare the function with a proper function prototype and include an error message if the file cannot be opened.

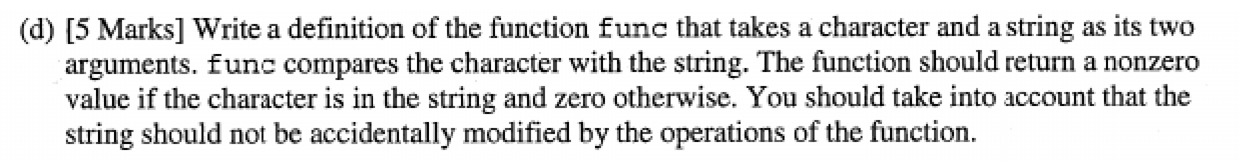
NWEN 241 FINAL EXAM 2014 Question 2



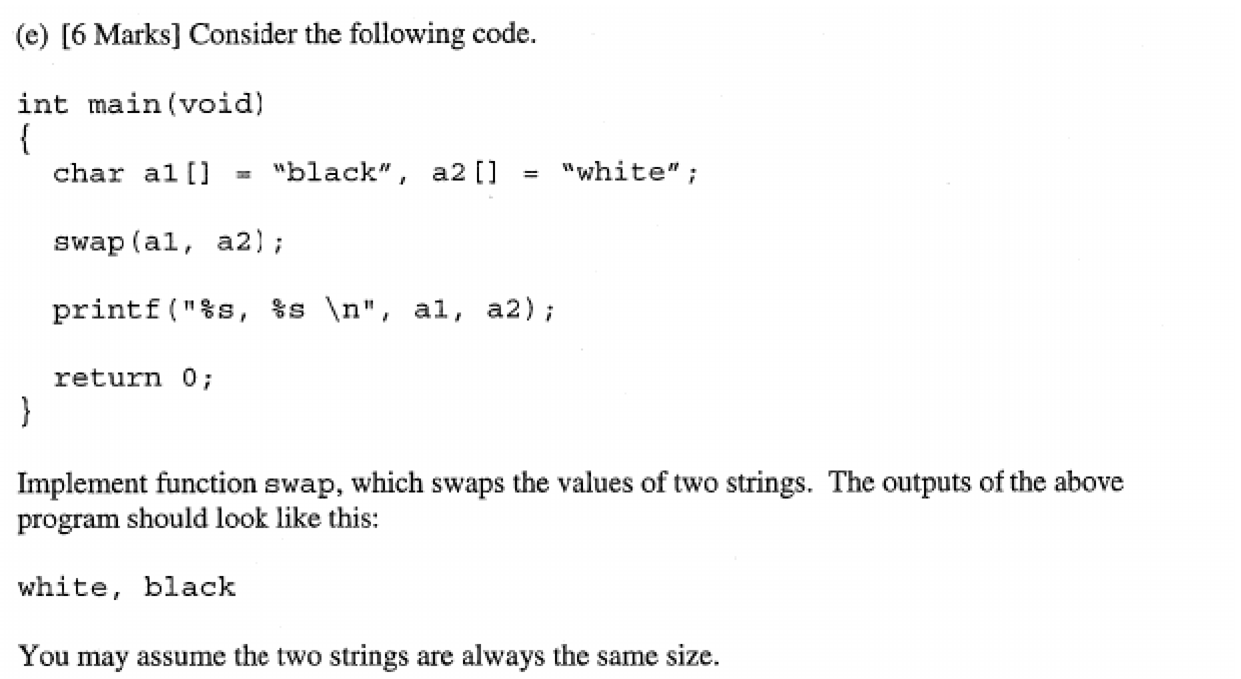
Question 2 Arrays and Pointers **[40 marks]**

(a) [5 Marks] Implement function `func`, with prototype `void func(char *)`, which will print out a sequence of suffixes of its argument in decreasing size. For example, if `func` was passed a string containing `"Monday"`, it would print out:

```
Monday, onday, nday, day, ay, y,
```



(d) [5 Marks] Write a definition of the function `func` that takes a character and a string as its two arguments. `func` compares the character with the string. The function should return a nonzero value if the character is in the string and zero otherwise. You should take into account that the string should not be accidentally modified by the operations of the function.



(e) [6 Marks] Consider the following code.

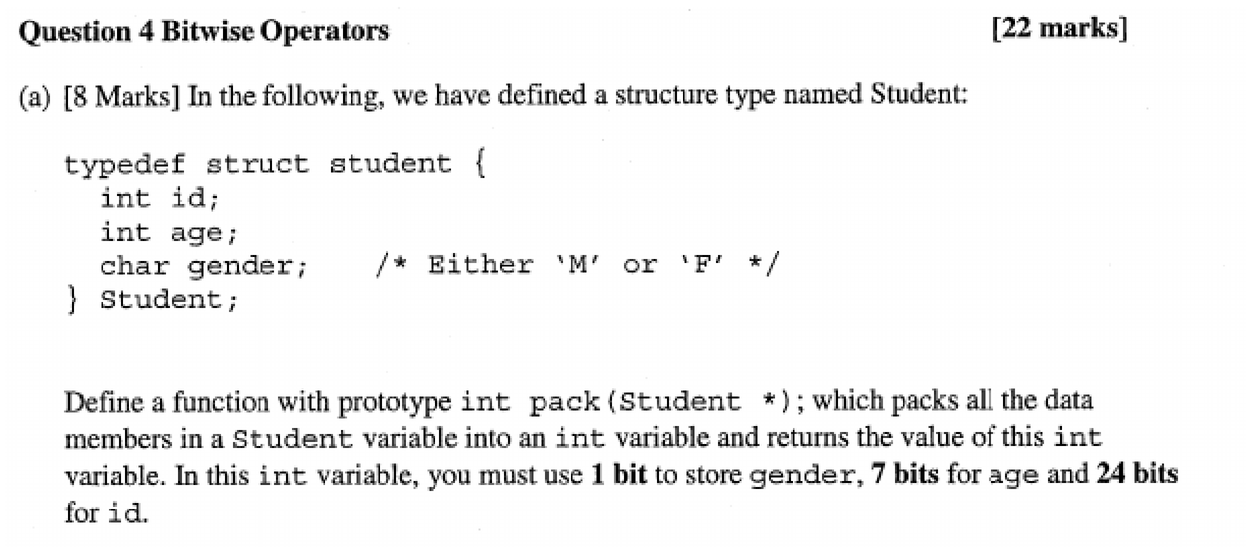
```
int main(void)
{
  char a1[] = "black", a2[] = "white";

  swap(a1, a2);

  printf("%s, %s \n", a1, a2);

  return 0;
}
```

Implement function `swap`, which swaps the values of two strings. The outputs of the above program should look like this:

```
white, black
```

You may assume the two strings are always the same size.

NWEN 241 FINAL EXAM 2014 Question 4



Question 4 Bitwise Operators **[22 marks]**

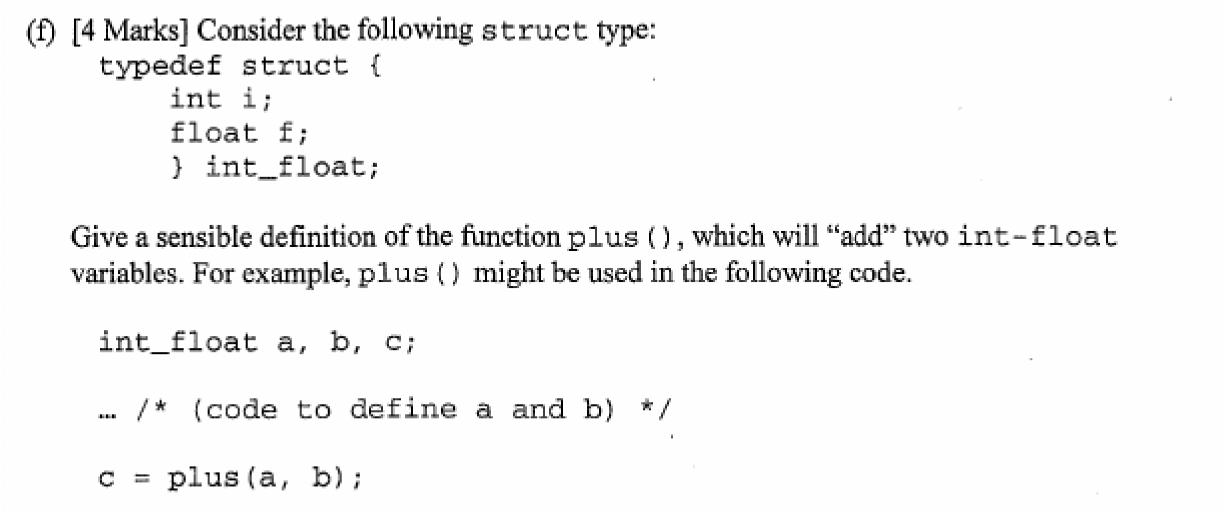
(a) [8 Marks] In the following, we have defined a structure type named Student:

```
typedef struct student {
  int id;
  int age;
  char gender;    /* Either 'M' or 'F' */
} Student;
```

Define a function with prototype `int pack(Student *);` which packs all the data members in a `Student` variable into an `int` variable and returns the value of this `int` variable. In this `int` variable, you must use **1 bit** to store `gender`, **7 bits** for `age` and **24 bits** for `id`.

(b) [6 Marks] Write a definition of the function `bitwise_swap` that uses only bitwise operators to swap the values of two integers.

NWEN 241 FINAL EXAM 2011 Question 4



(f) [4 Marks] Consider the following `struct` type:

```
typedef struct {
     int i;
     float f;
     } int_float;
```

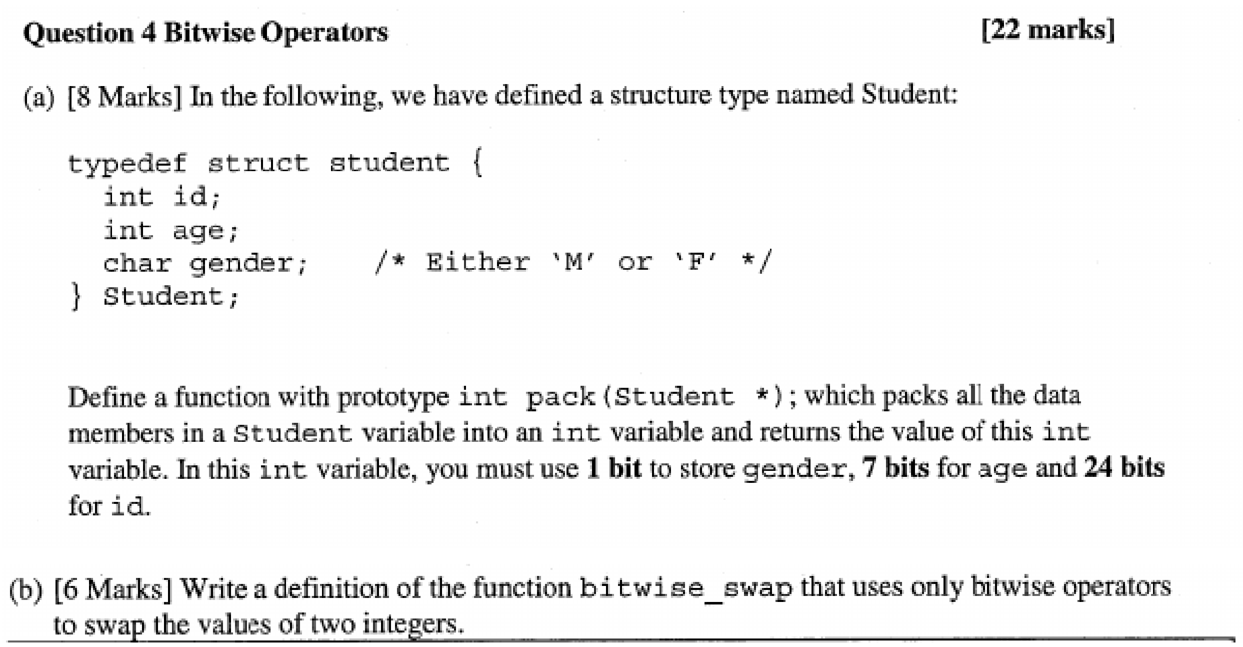
Give a sensible definition of the function `plus()`, which will “add” two `int-float` variables. For example, `plus()` might be used in the following code.

```
int_float a, b, c;

… /* (code to define a and b) */

c = plus(a, b);
```

(g) [3 Marks] State whether the following code is safe and justify your answer.

```
char *p;
strcpy(p, "test");
```

NWEN 241 FINAL EXAM 2011 Question 4



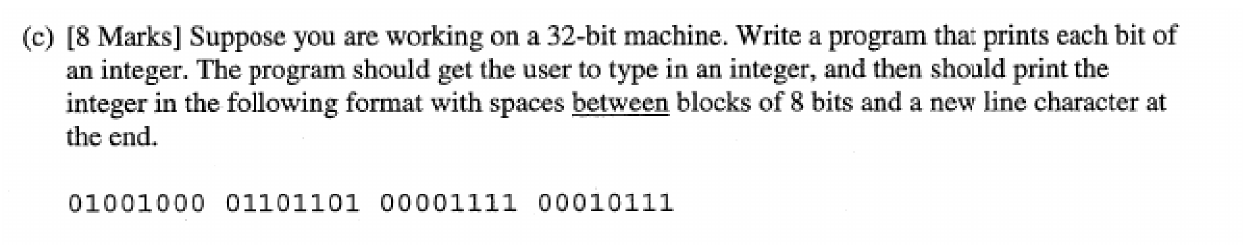
Question 4 Bitwise Operators **[22 marks]**

(a) [8 Marks] In the following, we have defined a structure type named Student:

```
typedef struct student {
  int id;
  int age;
  char gender;    /* Either 'M' or 'F' */
} Student;
```

Define a function with prototype `int pack(Student *);` which packs all the data members in a `Student` variable into an `int` variable and returns the value of this `int` variable. In this `int` variable, you must use **1 bit** to store `gender`, **7 bits** for `age` and **24 bits** for `id`.

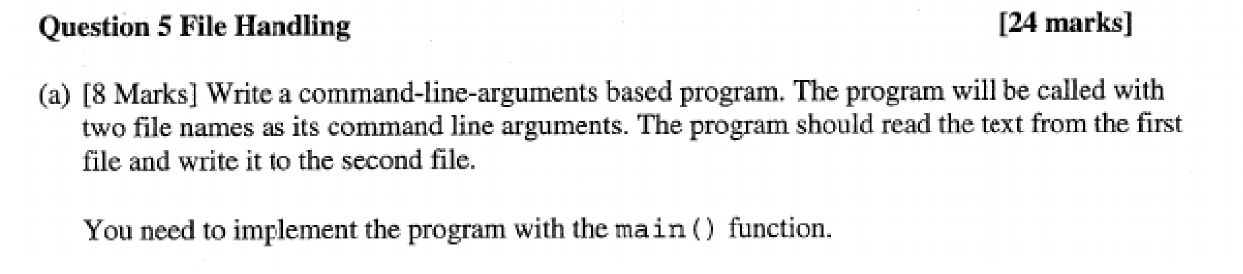
(b) [6 Marks] Write a definition of the function `bitwise_swap` that uses only bitwise operators to swap the values of two integers.



(c) [8 Marks] Suppose you are working on a 32-bit machine. Write a program that prints each bit of an integer. The program should get the user to type in an integer, and then should print the integer in the following format with spaces between blocks of 8 bits and a new line character at the end.

```
01001000 01101101 00001111 00010111
```

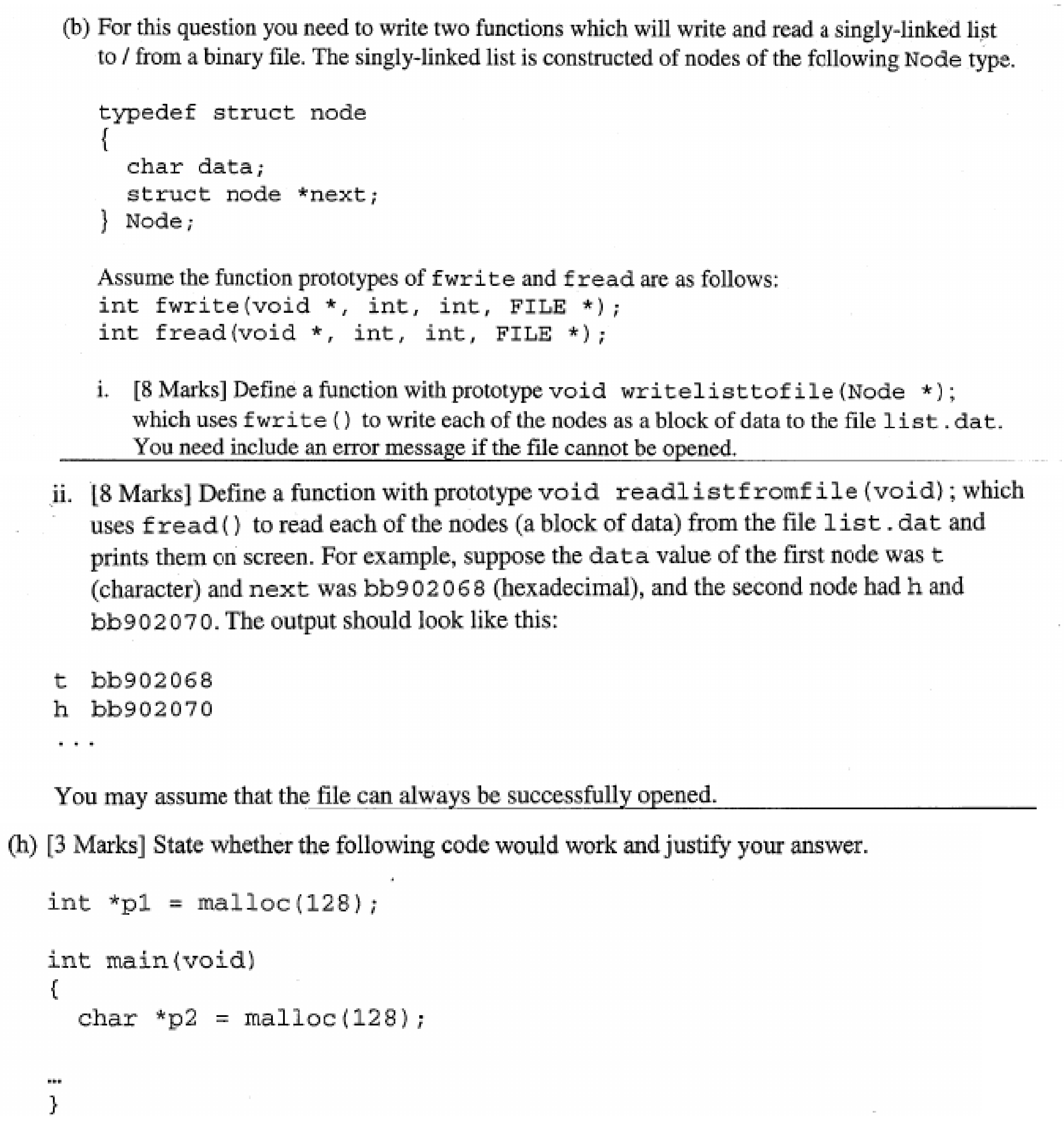
NWEN 241 FINAL EXAM 2011 Question 5



Question 5 File Handling **[24 marks]**

(a) [8 Marks] Write a command-line-arguments based program. The program will be called with two file names as its command line arguments. The program should read the text from the first file and write it to the second file.

You need to implement the program with the `main()` function.



(b) For this question you need to write two functions which will write and read a singly-linked list to / from a binary file. The singly-linked list is constructed of nodes of the following Node type.

```
typedef struct node
{
  char data;
  struct node *next;
} Node;
```

Assume the function prototypes of fwrite and fread are as follows:

```
int fwrite(void *, int, int, FILE *);
int fread(void *, int, int, FILE *);
```

i. [8 Marks] Define a function with prototype void writelisttofile(Node *); which uses fwrite() to write each of the nodes as a block of data to the file list.dat. You need include an error message if the file cannot be opened.

ii. [8 Marks] Define a function with prototype void readlistfromfile(void); which uses fread() to read each of the nodes (a block of data) from the file list.dat and prints them on screen. For example, suppose the data value of the first node was t (character) and next was bb902068 (hexadecimal), and the second node had h and bb902070. The output should look like this:

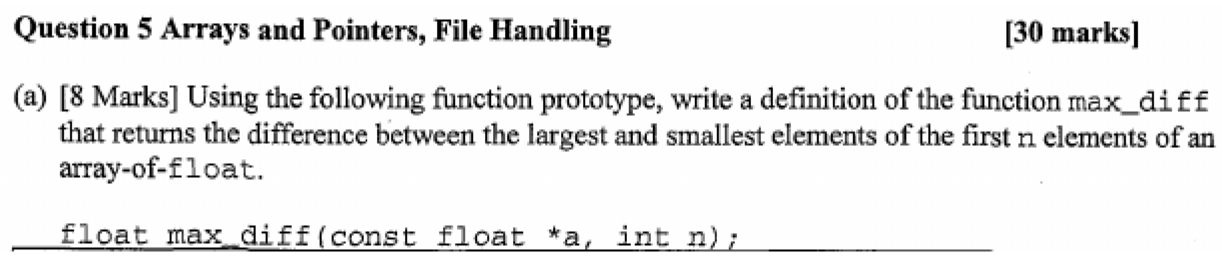
```
t bb902068
h bb902070
...
```

You may assume that the file can always be successfully opened.

(h) [3 Marks] State whether the following code would work and justify your answer.

```
int *p1 = malloc(128);

int main(void)
{
  char *p2 = malloc(128);

...
}
```

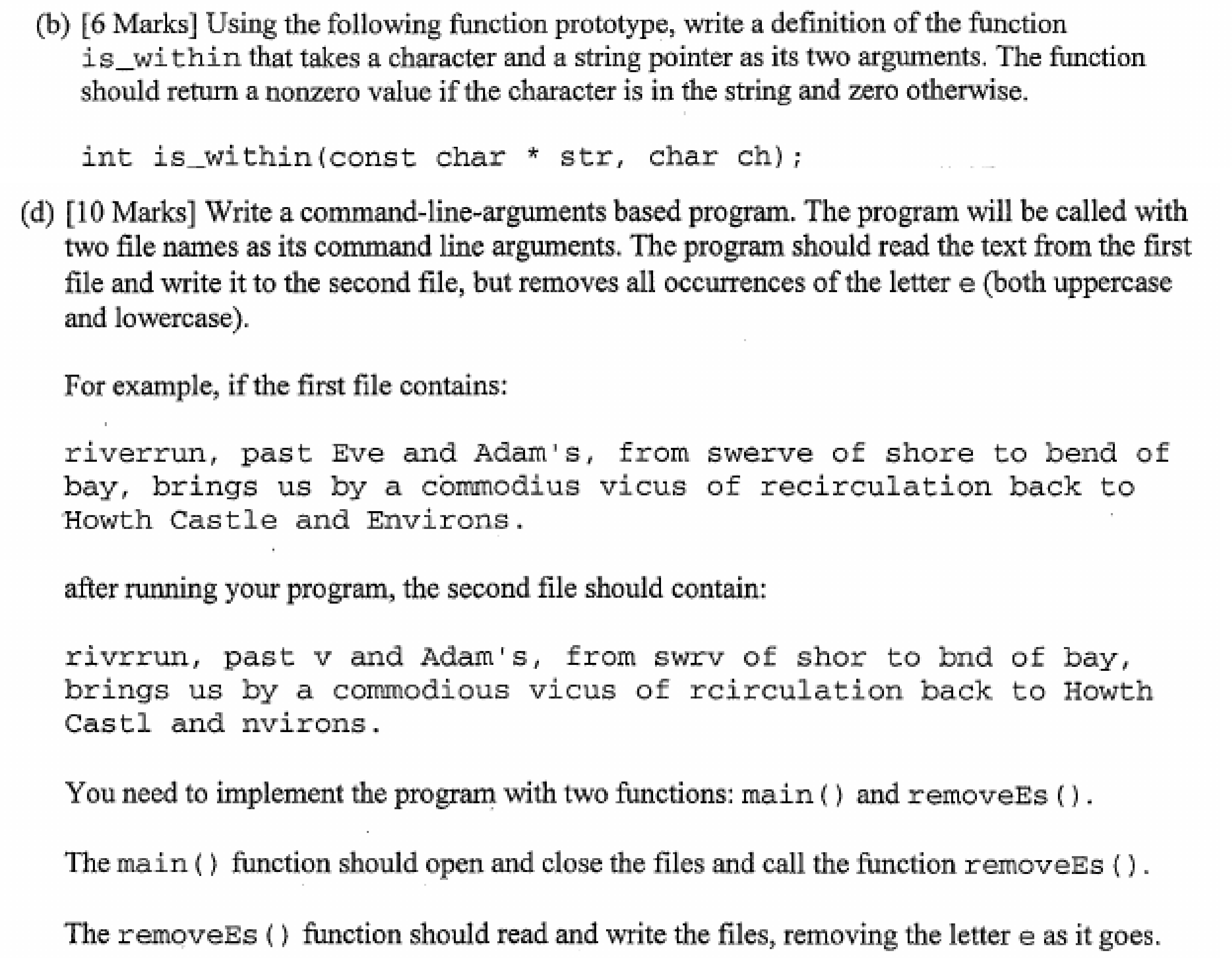
2011 Question 5



Question 5 Arrays and Pointers, File Handling **[30 marks]**

(a) [8 Marks] Using the following function prototype, write a definition of the function max_diff that returns the difference between the largest and smallest elements of the first n elements of an array-of-float.

```
float max_diff(const float *a, int n);
```



(b) [6 Marks] Using the following function prototype, write a definition of the function `is_within` that takes a character and a string pointer as its two arguments. The function should return a nonzero value if the character is in the string and zero otherwise.

```
int is_within(const char * str, char ch);
```

(d) [10 Marks] Write a command-line-arguments based program. The program will be called with two file names as its command line arguments. The program should read the text from the first file and write it to the second file, but removes all occurrences of the letter `e` (both uppercase and lowercase).

For example, if the first file contains:

```
riverrun, past Eve and Adam's, from swerve of shore to bend of
bay, brings us by a commodius vicus of recirculation back to
Howth Castle and Environs.
```

after running your program, the second file should contain:

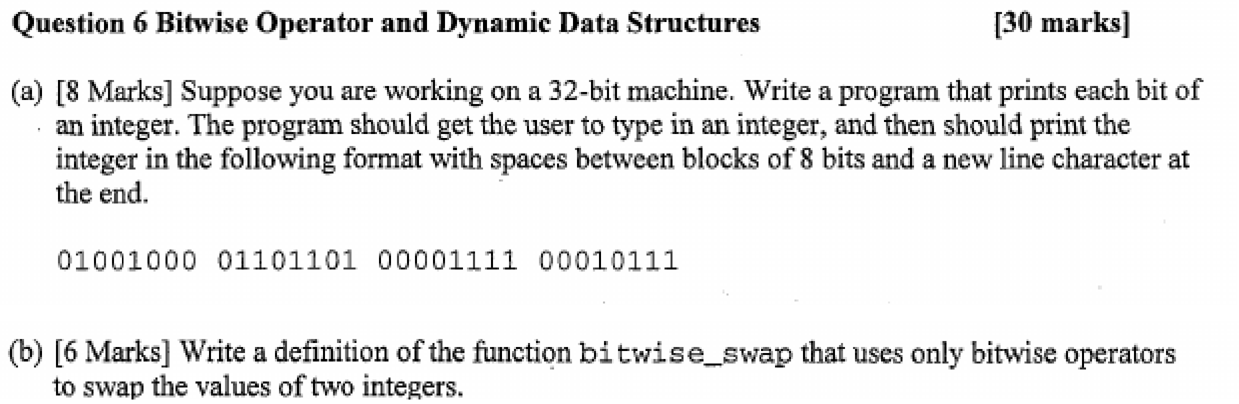
```
rivrrun, past v and Adam's, from swrv of shor to bnd of bay,
brings us by a commodious vicus of rcirculation back to Howth
Castl and nvirons.
```

You need to implement the program with two functions: `main()` and `removeEs()`.

The `main()` function should open and close the files and call the function `removeEs()`.

The `removeEs()` function should read and write the files, removing the letter `e` as it goes.

2011 Question 6



Question 6 Bitwise Operator and Dynamic Data Structures **[30 marks]**

(a) [8 Marks] Suppose you are working on a 32-bit machine. Write a program that prints each bit of an integer. The program should get the user to type in an integer, and then should print the integer in the following format with spaces between blocks of 8 bits and a new line character at the end.

```
01001000 01101101 00001111 00010111
```

(b) [6 Marks] Write a definition of the function `bitwise_swap` that uses only bitwise operators to swap the values of two integers.

(c) The following type definitions, macro definition and function prototypes are part of a queue model, where we are using singly-linked lists to implement queues.

```
#define Node_Size sizeof(Node)

typedef struct node
{ char data;
  struct node *next;
} Node;

typedef char Data;
typedef Node *ptrNode;

typedef struct queue
{ int cnt; /* counts the number of nodes */
  ptrNode front; /* points to the front node */
  ptrNode rear; /* points to the rear node */
} Queue;

typedef Queue *ptrQueue;

void enqueue(Data, ptrQueue);
Data dequeue(ptrQueue);
```

Suppose the queue that we have implemented has a header node of type `Queue` and a list of linked nodes of type `Node`. The pointer `front` in the header node points to the front node in the list, while the pointer `rear` points to the rear node in the list. The header node also has a counter `cnt`, which counts the number of nodes in the list.

i. [8 Marks] Write C code to implement the function `enqueue`, which adds a new node to the rear of the list. The character passed to `enqueue` needs to be assigned to the variable `data` in the node. You can assume your requests for memory are always successful. You DO NOT need to consider the case when the queue is empty and you may assume the queue is never full.

ii. [8 Marks] Write C code to implement the function `dequeue`, which deletes the front node from the list. The character stored in data needs to be returned. You DO NOT need to consider the case when the queue is empty, and you may assume the queue is never full.