

CS 135501 – Introduction to Programming

Midterm Exam: 3:20 – 5:10 p.m., November 14, 2005

If you do not understand a question, please raise your hand. Keep your eyes on your own paper and do not talk during the exam.

1. (10 points) Write TRUE or FALSE for each question.
 - (a) Like other high-level languages, C is generally considered to be machine independent.
 - (b) Comments in a C program cause a computer to print the text enclosed between `/*` and `*/` on the screen when the program is executed.
 - (c) C considers the variable **number** and **nUmBeR** to be identical.
 - (d) The **break** statement is required in the **default** case of a **switch** statement.
 - (e) An expression containing the `||` operation is true if at least one of its operands is true.
 - (f) An array can store many different types of values.
 - (g) An array subscript can be of data type **float**.
 - (h) A pointer that is declared to be **void** can be dereferenced.
 - (i) Two pointers that point to different arrays cannot be compared meaningfully.
 - (j) A sentinel value must be a value that cannot be confused with a legitimate data value.

2. (40 points) Pick the most appropriate choice for each question.
 - (a) Which of the following would not be considered as *hardware*?
 - (1) an operating system
 - (2) a CPU
 - (3) a keyboard
 - (4) a disk
 - (b) A computer can directly understand only its own _____.
 - (1) machine language
 - (2) assembly language
 - (3) high-level language
 - (4) none of the above
 - (c) Lines beginning with a `#` in a C program are processed
 - (1) at execution time
 - (2) at compile time
 - (3) at preprocessor time
 - (4) at postprocessor time
 - (d) Which of the following statements about the inclusion of `<stdio.h>` is false?
 - (1) It is required.
 - (2) This header file contains information and declarations used by the compiler when compiling standard input/output library functions such as **printf**.
 - (3) This header file contains information that helps the compiler determine if calls to library functions have been made correctly.
 - (4) This header file helps locate bugs in your program at compile time, rather than at execution time.

- (e) Which of the following is not a valid identifier?
- (1) **a_valid_identifier**
 - (2) **a2_valid_identifier**
 - (3) **a_valid_identifier_**
 - (4) **2_valid_identifier**
- (f) In a flowchart of an algorithm, what is the shape of a decision symbol?
- (1) circle
 - (2) rectangle
 - (3) diamond
 - (4) oval
- (g) What is wrong with the following loop?
- While (sum <= 1000)**
sum = sum + 30;
- (1) The parenthesis should be braces.
 - (2) Braces are required around **sum = sum +30;**.
 - (3) **While** should be **while**.
 - (4) There should be a semicolon after **While (sum <=1000)**.
- (h) If **x = 3**, which of the following sets **x** to 7?
- (1) **x *= 4;**
 - (2) **x += 4;**
 - (3) **x =+ 4;**
 - (4) **x + 4 = x;**
- (i) What is produced by a **for** statement with a correct body and with the following header?
- for (i = 20; i >= 2; i += 2)**
- (1) a syntax error
 - (2) a divide-by-zero error
 - (3) an infinite loop
 - (4) the even values of **i** from 20 down to 2.
- (j) Which expression raises **x** to the **y** power?
- (1) **x ** y**
 - (2) **x ^ y**
 - (3) **x pow y**
 - (4) **pow(x, y)**
- (k) Which data type should normally not be used to control a loop?
- (1) **int**
 - (2) **float**
 - (3) **short**
 - (4) **long**
- (l) A valid reason for building programs out of functions is
- (1) that the divide-and-conquer approach facilitates program construction
 - (2) that pre-existing functions can be used to create new programs
 - (3) the avoidance of code repetition within a program
 - (4) all of the above
- (m) Given the following function definition, the parameter list is represented by
- A B(C) {**

D

}

- (1) **A**
 - (2) **B**
 - (3) **C**
 - (4) **D**
- (n) What value does function **mystery** return when called with a value of 4?
- ```
int mystery (int number) {
 if (number <= 1)
 return 1;
 else
 return number * mystery(number - 1);
}
```
- (1) 1
  - (2) 24
  - (3) 0
  - (4) 4
- (o) Which definition tells the computer to reserve 12 elements for integer array **c**?
- (1) **c[ 12 ] int;**
  - (2) **int c [ 11 ];**
  - (3) **c[ 11 ] int;**
  - (4) **int c[ 12 ];**
- (p) Which of the following is true regarding the statement  
**++frequency[ responses[ answer ] ];**
- (1) This statement increases the appropriate frequency counter depending on the value of responses[ answer ].
  - (2) This statement increases the appropriate answer counter depending on the value of frequency[ responses ].
  - (3) This statement increases the appropriate responses counter depending on the value of frequency[ answer ].
  - (4) This statement produces a syntax error because subscripts cannot be nested.
- (q) A bubble sort of 1000 elements requires a maximum of \_\_\_\_\_ passes.
- (1) 1001
  - (2) 1000
  - (3) 999
  - (4) 998
- (r) Assume **Ptr** is a pointer variable. Which of the following values is different from the others?
- (1) **\*Ptr**
  - (2) **&\*Ptr**
  - (3) **\*Ptr**
  - (4) **Ptr**
- (s) An expression such as  
**sizeof(arrayName) / sizeof(double)**  
might typically be used to determine
- (1) the size of an array

- (2) the number of elements in an array
  - (3) the number of elements in half an array
  - (4) the size of an element of an array
- (t) Given that **k** is an integer array starting at location 2000, **kPtr** is a pointer to **k**, and each integer is stored in **4** bytes of memory, what location does **kPtr + 3** point to?
- (1) 2003
  - (2) 2006
  - (3) 2012
  - (4) 2024
3. (20 points) What is the output after executing each code segment or program?

(a)

```
int number[] = {10,8,6,4,2};
int *iptr, count;
iptr = &number[4];
*(iptr-3) = 100;
--iptr;
*iptr = 200;
*(iptr+1) = 300;
iptr = iptr - 3;
*iptr = 400;
for(count = 4; count >= 0; count--)
 printf("%d\n", number[count]);
```

(b)

```
#include <stdio.h>
void SomeFunction (int [], int);
int main ()
{
 int a[] = {1,2, 3, 4, 5};
 SomeFunction(a, 5);
 return 0;
}
void SomeFunction (int b[], int c)
{
 if (c > 0) {
 printf("%d\n", b[0]+c);
 SomeFunction(&b[1], c - 1);
 }
}
```

(c)

```
#include <stdio.h>
void f(int, int);
int main() {
 int a=12, b=5;
```

```

 f(b,a);
 printf("a=%d b=%d\n",a,b);
 return 0;
}
void f(int a, int b) {
 b = 3*a;
 a = (b%5 - 1)*b;
 printf("a=%d b=%d\n",a,b);
}

```

(d)

```

#include <stdio.h>
void f(int *, int *);
int main() {
 int a=5, b=12;
 f(&b,&a);
 printf("a=%d b=%d\n",a,b);
 return 0;
}
void f(int *a, int *b) {
 b = 3(*a);
 *a = ((*b)%5 - 1)*(*b);
 printf("a=%d b=%d\n",*a,*b);
}

```

4. (10 points)

- (a) Rewrite the following program segment using a **for** statement. In the body of your **for** statement, replace the conditional operator (?:) with a **if...else** statement.

```

while (-- counter >= 1)
 printf("%s\n", counter % 2 ? "even" : "odd");

```

- (b) Rewrite the following program segment using a series of **if** statements. (You cannot use any **if...else** statement.)

```

switch (grade) {
 case 'A':
 printf("Outstanding grade");
 break;
 case 'B':
 case 'C':
 printf("Good grade");
 break;
 default:
 printf("Bad grade");
 break;
}

```

5. (10 points) Write a program to read in two positive integers **M** and **N** from the keyboard, call a function **gcd** taking **M** and **N** as the arguments to calculate the greatest common divisor of **M** and **N**, and print out the result. The greatest common divisor of **M** and **N** is the largest integer that divides both **M** and **N**.
6. (10 points) Write a recursive function **StringReverse** that takes a character array as the only argument, prints the array elements back to front, and returns nothing. Note that the last element of a character array is always the **NULL** character, i.e., '\0'.