CENG 140
Spring '15
Midterm Sample

• About this sample:
  – This sample does not provide an indication of the actual question count nor subject
distribution.
  – The sample does not restrict the subjects you are responsible of. In other words in
the midterm you may have question from subject which this sample has not covered
but was among the announced midterm subjects.
  – Don’t expect questions that are similar or variants of the sample questions to appear
in the exam.

• About the exam questions:
  – Unless it is stated explicitly, all questions are to be answered in ANSI C context.
  – Furthermore, you shall not concentrate on the general lay out of the program, nor
make assumptions that the undisplayed lines may be a source of error. In the
questions the answer will only and only rely on the information in the displayed part.
  – All string constants are modifiable.
  – Assume that all the necessary ANSI header files are properly included and this is not
a source of error.
  – If you need, assume:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>sizeof(int)</td>
<td>4</td>
<td>sizeof(float)</td>
</tr>
<tr>
<td>sizeof(char *)</td>
<td>4</td>
<td>sizeof(long)</td>
</tr>
<tr>
<td>sizeof(double)</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>
We start with TRUE-FALSE questions, mark A box for TRUE, B box for FALSE on your answer sheet

1  Macro replacements are done at run-time.

2  Function calls to functions that return void, cannot be used in the right hand side of an assignment.

3  By the ANSI standard of C, you can omit the test expression in a while loop like you can omit the test expressions of a for loop, to have an infinite loop.

4  All l-values are also r-values.

5  C implements the concept of so called watchdog expressions. To understand this, a good example is the way do...whiles are evaluated. The conditional expression here in is compiled as a ‘watchdog expression’. By this, if the conditional expression becomes 0 (false) in the middle of the loop body, a break is executed and the loop is exited immediately.

6  In a prototype line of a function
   • the data type and the order of the parameters
   • the data type of the return value

   are defined. Having to hand this information the compiler is able to structure the lay out of the activation record. Is this true?

7  (float)a/b produces the same result as a/(float)b for int variables a and b.

8  char *p="ARLUT", *q="ARNUT";
\main()
{ p[2]++; *(q+5/2) += p[2]-q[2];
  printf("%d", p==q); }

will print 0, is this true?

9  "ARMUT"=="ARMUT"

   Has a value of 1 because string literals are stored uniquely in C. This property is also known as SUS (String Unification Scheme), defined in the ANSI standard leaving the details of the Scheme undefined for C compiler implementors.

10 A switch body without any break will cause a compile-time error. A switch has to contain at least one break. Mostly this is achieved by the programmer by putting an unnecessary break after the default.

11 #define INT_PTR int *
INT_PTR ptr1 , ptr2;

will declare ptr1 and ptr2 to be two int pointers.

The TRUE-FALSE questions ENDED here.
12 int i,*ip;
main()
{ i=5;
 ip=&i;
 *ip=&ip;
 printf("%d",i); }
The given program will
 A) print nothing.       D) produce a run-time error.
 B) print 5                  E) None of these
 C) produce a compile-time error.

13 void f(char *cp)
{ if (*cp==cp[-1]) *cp++; }
void main()
{ char c[]="hello";
   int i;
   for (i=1; i<6; i++) f(c+i);
   printf("%s",c); }
The given program will
 A) print helmo
 B) print h
 C) print hello
 D) produce a compile-time error (array index cannot be negative).
 E) produce a run-time error.

14 int a[5]={2,4,6,8,10}, b[5]={0,0,0,0,0};
int i;
main ()
{ for (i=4; i; a[i]+=i--);
   for ( ; i<5; b[i]=a[i]/(i++));
   for (i=0; i<5; i++) printf("%d ",b[i]); }
will
 A) print 0 2 2 2 2
 B) print 0 2 2 0
 C) print 2 2 2 2
 D) produce a run-time error (division by zero).
 E) produce a compile-time error (missing expression in for construct).

15 int x,y;
main()
{ for (x=0, y=5; --y && ((++x || y--) || (x=y)); ) ; }
What is the value of x after the for statement is executed.
 A) 1       D) 5
 B) 2       E) Code will cause a compile-time error.
 C) 4
float root_1(float b, float sqr_disc, float a)
{ return ( (-b+sqr_disc)/2*a ); }

main()
{ printf("%f\n", root_1( 0, 24, 4)); }
The given piece of code will
A) produce a run-time error
B) produce a compile-time error
C) print 3.000000
D) print -3.000000
E) None of these

int round(int *x)
{ if (!(*x++ % 3)) return *x - 1;
else if ( !(*x++ % 3)) return *x - 1;
else return *x; }
main()
{ int x=6, i;
for (i=0; i<3; i++, x++)
{ printf("x=%d",x);
printf(" %d\n", round(&x)); } }
Will print
A) x=6  6
   x=8  9
   x=11 12
B) x=6  5
   x=8  5
   x=11 15
C) x=6  6
   x=7  9
   x=8  9
D) x=6  5
   x=7  8
   x=8  8
E) None of these

int fun(int a, int b, int c)
{ if (a < b < c)
   printf("quot = %d\n", b-a / c-b );
else if (a < b)
   printf("c-b may be zero! %d\n", b-a / c-b ); }
main()
{ fun(-11, 5, 3); }
will
A) output quot = -8
B) output quot = 4
C) output quot = 3
D) output c-b may be zero! -8.
E) produce a compile-time error (illegal syntax of <)