CS 241 Data Organization Quiz 2

Brooke Chenoweth

University of New Mexico

Fall 2014

Question 1: Syntax

Which of these statements would result in an error?

A `int a = 40 * 2*(1 + 3);`

B `int b = (10 * 10 * 10) + 2`

C `int c = (2 + 3) * (2 + 3);`

D `int d = 1/2 + 1/3 + 1/4 + 1/5 + 1/6;`

E `int e = 1/2 - 1/4 + 1/8 - 1/16;`

Question 1: Syntax

Which of these statements would result in an error?

A `int a = 40 * 2*(1 + 3);`

B `int b = (10 * 10 * 10) + 2`

C `int c = (2 + 3) * (2 + 3);`

D `int d = 1/2 + 1/3 + 1/4 + 1/5 + 1/6;`

E `int e = 1/2 - 1/4 + 1/8 - 1/16;`

Question 2: ‘=’ symbol

In the C programming language, the ‘=’ symbol is most accurately read:

A “Equals”

B “Assign the value of the expression on the right side to the variable on the left”

C “Is equivalent to”

D “A mathematical symbol used to indicate equality”

E “A conditional symbol used to indicate equality”

Question 2: ‘=’ symbol

In the C programming language, the ‘=’ symbol is most accurately read:

A “Equals”

B “Assign the value of the expression on the right side to the variable on the left”

C “Is equivalent to”

D “A mathematical symbol used to indicate equality”

E “A conditional symbol used to indicate equality”

Question 3: Alphabet char

```
char c = getchar();
```

Which is true *if and only* if c is a letter in the standard English alphabet?

A `((c>='a'&& c<='z') && (c>='A'&& c<='Z'))`

B `((c>='a'&& c<='z') || (c>='A'&& c<='Z'))`

C `((c>='a'|| c<='z') || (c>='A'|| c<='Z'))`

D `((c>='a'|| c<='z') && (c>='A'|| c<='Z'))`

E `( c>='a'&& c<='z' && c>='A'&& c<='Z')`

Question 3: Alphabet char

```
char c = getchar();
```

Which is true *if and only* if `c` is a letter in the standard English alphabet?

A `((c>='a'&& c<='z') && (c>='A'&& c<='Z'))`

B `((c>='a'&& c<='z') || (c>='A'&& c<='Z'))`

C `((c>='a'|| c<='z') || (c>='A'|| c<='Z'))`

D `((c>='a'|| c<='z') && (c>='A'|| c<='Z'))`

E `( c>='a'&& c<='z' && c>='A' && c<='Z')`

Question 4: Call by value

In the C Programming Language, *call by value* means:

A When two functions have the same name, the compiler determines which to call by the value of the arguments.

B The called function is given the address of its arguments so that the function can both read and set the arguments values.

C Each called function is assigned a value that is used by the operating system to determine the functions priority. This is most useful on multi-core systems.

D The called function is given the values of its arguments which are copied into temporary variables.

Question 4: Call by value

In the C Programming Language, *call by value* means:

A When two functions have the same name, the compiler determines which to call by the value of the arguments.

B The called function is given the address of its arguments so that the function can both read and set the arguments values.

C Each called function is assigned a value that is used by the operating system to determine the functions priority. This is most useful on multi-core systems.

D The called function is given the values of its arguments which are copied into temporary variables.

Question 5: Automatic variable

In the C Programming Language, an *automatic variable* is:

A A local variable in a function which comes into existence at the time the function is called, and disappears when the function is exited.

B A variable that is automatically initialized.

C A global variable that is automatically available to all functions within the source file.

D A global variable that is available to all functions within any source file that declare the variable as extern.

E A variable that is automatically defined by the compiler such as PI, E, and HBAR.

Question 5: Automatic variable

In the C Programming Language, an *automatic variable* is:

A A local variable in a function which comes into existence at the time the function is called, and disappears when the function is exited.

B A variable that is automatically initialized.

C A global variable that is automatically available to all functions within the source file.

D A global variable that is available to all functions within any source file that declare the variable as extern.

E A variable that is automatically defined by the compiler such as PI, E, and HBAR.

Question 6: `if`, `else if`, `else`

```c
int main(void)
{
  int x = 4;

  if (x == 1)
  {
    printf("x is 1\n");
  }
  else if (x == 2)
  {
    printf("x is 2\n");
  }
  else x = 3;
  {
    printf("x is %d\n", x);
  }
}
```

What is the output of this code?

A `x is 1`

B `x is 2`

C `x is 3`

D `x is 4`

E Nothing is printed.

Question 6: `if`, `else if`, `else`

```
int main(void)
{
  int x = 4;

  if (x == 1)
  {
    printf("x is 1\n");
  }
  else if (x == 2)
  {
    printf("x is 2\n");
  }
  else x = 3;
  {
    printf("x is %d\n", x);
  }
}
```

What is the output of this code?

A `x is 1`

B `x is 2`

C `x is 3`

D `x is 4`

E Nothing is printed.

Question 7: Flag

```
int main(void)
{
  int i, n;
  int flag;
  for (n=10; n>1; n--)
  {
    flag = 0;
    for (i=2; i<n; i++)
    {
      if(n % i == 0) flag = 1;
    }
    if (flag == 0) printf("%d ", n);
  }
  printf("\n");
}
```

What is the output of this code?

A 10 9 8 7 6 5 4 3 2

B 9 8 7 6 5 4 3

C 9 7 5 3

D 7 5 3 2

Question 7: Flag

```c
int main(void)
{
  int i, n;
  int flag;
  for (n=10; n>1; n--)
  {
    flag = 0;
    for (i=2; i<n; i++)
    {
      if(n % i == 0) flag = 1;
    }
    if (flag == 0) printf("%d ", n);
  }
  printf("\n");
}
```

What is the output of this code?

A 10 9 8 7 6 5 4 3 2

B 9 8 7 6 5 4 3

C 9 7 5 3

D 7 5 3 2

Aside: Flag

- In computer programming, *flag* often refers to a variable or bit used to indicate a particular property is “on” or “off”.
- Generally a good idea to use a more meaningful name than “flag”, though.

Aside: More efficient printer of primes

```
int main(void)
{
  int i, n;
  int flag;
  for (n=10; n>1; n--)
  {
    flag = 0;
    for (i=2; i<n; i++)
    {
      if(n % i == 0)
      {
        flag = 1;
        break;
      }
    }
    if (flag == 0) printf("%d ", n);
  }
  printf("\n");
```

Once a factor of `n` is found, `n` cannot be prime, so break out of inner loop.

Question 8: Functions

This code will not compile because:

```
1  int foo(float x);
2
3  void main(void)
4  {
5     int n=5;
6     printf("%d\n", foo(n));
7  }
8
9  int foo(int n)
10 {
11    return 2*n;
12 }
```

A The version of foo in line 1 accepts a float, but returns an int.

B The function foo in line 1 has no body.

C The version of foo in line 1 should not end with a semicolon.

D The variable n is declared in two different places.

E The prototype of foo does not agree with the definition.

Question 8: Functions

This code will not compile because:

```c
 1  int foo(float x);
 2
 3  void main(void)
 4  {
 5     int n=5;
 6     printf("%d\n", foo(n));
 7  }
 8
 9  int foo(int n)
10  {
11     return 2*n;
12  }
```

A The version of foo in line 1 accepts a float, but returns an int.

B The function foo in line 1 has no body.

C The version of foo in line 1 should not end with a semicolon.

D The variable n is declared in two different places.

E The prototype of foo does not agree with the definition.