Pointer Assessment Quiz – Version 0.5[1]

This is a **preliminary version** of this assessment instrument and has not yet been piloted. Items are presented according to learning outcomes for outcomes A-E. Items have yet not been developed for Outcomes F, G, and H. We have also skipped Outcomes C, since it doesn't seem to be fundamental to pointers.

The legend for the type of question posed is C for Concepts/Definition Questions, T for Tracing, and W for Writing.

For the code tracing questions, we note that all of these questions assume knowledge of `iostream`, using namespace `std`, and code inside a main function. Students *may* think that the entire quiz is nothing but "gotchas" and answer none of the above on every question. We could make all of the questions more verbose by including the elided code, or just assume none one would make that assumption.

We also note that Learning Outcome C does not seem to be fundamental to pointers, and therefore no items are included in this version of the assessment.

Learning Outcome A: Value Basics, No Pointers

These questions are fundamentally basic. We include them here, since we have a learning outcome for values and they might be useful by providing a mechanism to remove a participant from the study. If they can't answer these questions correctly then they can't be expected to learn pointers.)

(C) Question 1 (modifying variables)

For the following snippet of code, select all statements that are true about line 3.

```
1 int a = 1;
2 int b = 2;
3 a = b + 1;
4 cout << a << " " << b;
```

A. line 3 will change the value of the variable a when executed (*)
B. line 3 will change the value of the variable b when executed
C. line 3 will evaluate to false because 1 does not equal 3
D. line 3 will prevent the program from compiling because 1 does not equal 3

(C) Question 2 (arrays)

For the following snippet of code, which lines will produce compile-time errors?

```
1 int a = 1;
2 int b;
3 char c = 'A';
4 char d;
5 a = b;
6 a = c;
```

[1] This is an early, unpiloted version of this assessment instrument. If you use this instrument in any way, please credit the authors of the corresponding paper: <Paper citation will go here.>

```
7 b = d;
8 d = c;
```

(Note: This question might be a little too tricky. Since C++ is so permissive, none of the lines produce a compile-time error.)

(T) Question 3 (assign from literal, copy variable value, overwrite value)

What would the following snippet of code print when run?

```
char a;
char b;
a = 'B';
b = a;
a = 'A';
cout << a << " " << b;
```

A. A A
B. A B (*)
C. B A
D. B B
E. None of the above

(T) Question 4 (arrays)

What would the following snippet of code print when run?

```
char a[2];
char b;
a[0] = 'A';
a[1] = 'B';
a[0]= a[1];
a[1] = a[0];
cout << a[0] << " " << a[1];
```

A. A A
B. A B (*)
C. B A
D. B B
E. None of the above

(W) Question 5

For the following snippet of code,

```
1 int a;
2 cin >> a;
3
4 cout << a;
```

which of the following lines, if placed in line 3, would create a program that prints the number 1 greater than what the user enters?

A. a + 1;
B. a = a + 1; (*)

C. a++; (*)

Learning Outcome B: Pointer Basics

(C) Question 1 (definitions)

(This question as shown below are matching or multiple choice with multiple correct answers. To make the value of questions more consistent these should probably be modified to be multiple choice with a single answer.)

For the following snippet of code, match the line number to the description of the operation performed on that line.

```
1 char a;
2 char *p;
3 a = 'A';
4 p = &a;
5 cout << *p;
```

A. pointer declaration (2)
B. pointer dereference (5)
C. pointer reference (4)

(T) Question 2 (copy pointer, dereference in print)

What would the following snippet of code print when run?

```
char *p;
char *q;
p = new char('B');
q = p;
p = new char('A');
cout << *p << " " << *q;
```

A. A A
B. A B (*)
C. B A
D. B B
E. None of the above

(T) Question 3 (deference pointer in assignment to value)

What would the following snippet of code print when run?

```
char *p;
char a;
p = new char('A');
a = *p;
p = new char('B');
cout << *p << " " << a;
```

A. A A
B. A B
C. B A (*)
D. B B
E. None of the above

(T) Question 4 (reference value in assignment to pointer)

What would the following snippet of code print when run?

```
char a;
char *p;
a = 'A';
p = &a;
a = 'B';
cout << a << " " << *p;
```

A. A A
B. A B
C. B A
D. B B (*)
E. None of the above

(T) Question 5 (value assigned to dereferenced pointer)

What would the following snippet of code print when run?

```
char a;
char *p;
a = 'A';
p = new char('B');
*p = a;
cout << a << " " << *p;
```

A. A A (*)
B. A B
C. B A
D. B B
E. None of the above

(T) Question 6 (dereferenced pointer assigned to dereferenced pointer)

What would the following snippet of code print when run?

```
char *p;
char *q;
p = new char('A');
q = new char('B');
*q = *p;
cout << *p << " " << *q;
```

A. A A (*)
B. A B
C. B A
D. B B
E. None of the above

(W) Question 7 (assign pointer to pointer)

The following snippet of code is incomplete:

```
char *p;
char *q;
p = new char('A');
q = XXX;
cout << *q;
```

Which of the following in place of XXX will complete the program so that it prints A when run.

A. p (*)
B. *p
C. &p
D. None of the above

(W) Question 8 (assign pointer dereference to value)

The following snippet of code is incomplete:

```
char *p;
char a;
p = new char('A');
a = XXX;
cout << a;
```

Which of the following in place of XXX will complete the program so that it prints A when run.

A. p
B. *p (*)
C. &p
D. None of the above

(W) Question 3 (assign value reference to pointer)

The following snippet of code is incomplete:

```
char *p;
char a;
a = 'A';
p = XXX;
cout << *p;
```

Which of the following in place of XXX will complete the program so that it prints A when run.

A. a
B. *a
C. &a (*)
D. None of the above

(W) Question 4 (assign value to pointer dereference)

The following snippet of code is incomplete:

```
char *p;
char a;
p = new char('B');
a = 'A';
*p = XXX;
cout << *p;
```

Which of the following in place of XXX will complete the program so that it prints A when run.

A. a (*)
B. &a
C. *a
D. None of the above

(W) Question 5 (assign pointer dereference to pointer dereference)

The following snippet of code is incomplete:

```
char *p;
char *q;
p = new char('A');
q = new char('B');
*q = XXX;
cout << *q;
```

Which of the following in place of XXX will complete the program so that it prints A when run.

A. p
B. &p
C. *p (*)
D. None of the above

Learning Outcome D (Values Vs Pointers AKA Common Pointer Errors)

The tracing section of value's vs pointers was common mistakes. I feel like those common mistakes are already covered by the above questions where the common mistakes are the incorrect options.

(C) Question 1 (value vs pointer)

For each variable in the following snippet of code, specify whether the memory for the variable contains a char value or an address that refers to a location in memory that contains a char value.

```
char a = 'A';
char *p = &a;
char b = *p;
char *q = p;
```

- a (value)
- p (address)
- b (value)
- q (address)

(C) Question 2 (errors)

For the following snippet of code, which lines will produce compile-time errors?

```
1  char a = 'A';
2  char b = 'B';
3  char *p = new char('C');
4  char *q = new char('D');
5  a = b;
6  p = &a;
7  b = *q;
8  p = q;
9  a = p;  (*)
10 q = b;  (*)
11 p = *a; (*)
12 b = &q  (*)
```

(T) Question 4 (forget to dereference pointer in print)

What would the following snippet of code print when run?

```
char *p;
char *q;
p = new char('A');
q = p;
p = new char('B');
cout << p << " " << q;
```

A. A A
B. A B
C. B A

D. B B
E. None of the above (*)

(Note, this code actually produces a run-time error instead of printing out memory addresses because cout thinks p and q are C-strings without the terminating null character. Could change this to ints, but then it would stick out as the only example with ints, so maybe change all of them to use ints? Or replace it with some other example of forgetting to dereference a pointer? I don't know.)

(T) Question 5 (forget to dereference pointer in assignment)

What would the following snippet of code print when run?

```
char *p;
char a;
p = new char('A');
a = p;
p = new char('B');
cout << *p << " " << a;
```

A. A A

B. A B

C. B A

D. B B

E. None of the above (*) (Note, this is a compile-time error for incompatible types. The question prompt may need to be rephrased to allow for it not being possible to even run the code, or perhaps adding an additional choice for compile-time error and/or run-time error)

(T) Question 6 (forget to dereference pointer or reference value in assignment)

What would the following snippet of code print when run?

```
char a;
char *p;
a = 'A';
p = a;
a = 'B';
cout << a << " " << *p;
```

A. A A
B. A B
C. B A
D. B B
E. None of the above (*) (compile-time error for incompatible types)

Learning Outcome E (Pointer Arithmetic)

(C) Question 1 (array increment)

For the following snippet of code, what does line 3 do?

```
1 int *p = new int[2] {1, 2};
3 p++;
4 cout << *p;
```

A. Changes the address of the pointer p.

B. Changes the value that the pointer p refers to.

C. Neither of the above, it is an error.

(T) Question 2 (pointer increment)

What would the following snippet of code print when run?

```
char *p;
char a;
p = new char[2] {'A', 'B'};
a = *p;
p++;
*p = a;
cout << *p << " " << a;
```

A. A A (*)

B. A B

C. B A

D. B B

E. None of the above

(T) Question 3 (pointer offset)

(Note, I'm not sure this question is appropriate because it depends on chars being 1 byte in size, not something that has necessarily come up in every class when pointers are covered.)

What would the following snippet of code print when run?

```
char *p;
char a;
p = new char[2] {'A', 'B'};
a = *(p + 1)
*p = a;
cout << *p << " " << a;
```

A. A A (*)

B. A B

C. B A

D. B B

E. None of the above

(T) Question 4 (pointer offset out of bounds)

(Note, this might be considered more of an array question than a pointer question, but pointer arithmetic error is included in the learning outcomes.)

What would the following snippet of code print when run?

```
char *p;
char a;
p = new char[2] {'A', 'B'};
a = *(p + 2)
*p = a;
cout << *p << " " << a;
```

A. A A

B. A B

C. B A

D. B B

E. None of the above (*) (I got a run-time error, I thought it would print garbage. Perhaps it depends on your system, or I need to learn C++ better.)

(T) Question 5 (pointer offset not out of bounds)

(Note, this might be considered more of an order of operations question.)

What would the following snippet of code print when run?

```
char *p;
char a;
p = new char[2] {'A', 'B'};
a = *p + 2;
*p = a;
cout << *p << " " << a;
```

A. A A

B. A B

C. B A

D. B B

E. None of the above (*) (It prints 'C'.)

(W) Question 6 (pointer increment)

The following snippet of code is incomplete:

```
char *p;
p = new char[2] {'A', 'Z'};
p = XXX;
cout << *p;
```

Which of the following in place of XXX will complete the program so that it prints Z when run.

A. p + 1 (*)

B. *p + 1

C. *(p + 1)

D. None of the above

(W) Question 7 (pointer offset)

The following snippet of code is incomplete:

```
char *p;
char a
p = new char[2] {'A', 'Z'};
a = XXX;
cout << a;
```

Which of the following in place of XXX will complete the program so that it prints Z when run.

A. p + 1

B. *p + 1

C. *(p + 1)

D. None of the above