CPA Chapter 3 Practice Quiz

C++ Institute Volunteer Program 2015

AUTHOR:	ADRIAN NECULA	ADRIAN.NECULA@LIVE.COM
AUTHOR'S BIO:	I am working as a C/C++ programmer at Siemens.	

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1	Pointers: another kind of data in the “C++” language		
C++ Associate (CPA)	Chapter: 3	Section: 1	Question type: Single-choice
Subject: Pointer Basic Information			Question Number: 1

Question: What is the output of the following code fragment in C++ ? (assumption: all #include and the rest of the code are correct)

```
int myInt1 = 3, myInt2 = 3;
int *pnt1 = &myInt1, *pnt2 = &myInt2;
myInt1 = ++(*pnt1) + (*pnt1);
myInt2++;
myInt2 = (*pnt2) + (*pnt2);
cout<< myInt1<<myInt2<<endl;
```

Answers:

A) 86
B) 78
C) 88
D) 77

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1-6,10			
C++ Associate (CPA)	Chapter: 3	Section: 1-6,10	Question type: Single-choice
Subject: Pointers, functions, and memory management			Question Number: 2

Question: What is the output of the following code fragment in C++

```
void SetElements( int index, int **array, int value = 0){
   (*array)[*(&index)] =  value;
};
Int main(){
   int *point1, *point2;
   point1 = new int[1];
   point2 = new int[2];
   *point1 = 0;
   SetElements(*&*point1, &point2);
   point1[0] = 1;
   SetElements(*&*point1, &point2, *point1 );
   cout<< point2[(*point1)]<<point2[(*point1)-1]<<endl;
  delete[] point1;
  delete[] point2;
}
```

Answers:

A) 11
B) 10
C) 01
D) 00

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 3,6			
C++ Associate (CPA)	Chapter: 3	Section: 3,6	Question type: Multiple-choice
Subject: Functions overloading			Question Number: 3

Question: Which of the function call will generate a compiler error?

```
void f(){}
void f(int){}
void f(float){}
void f ( float, int = 0){}
int main(){
   short myShort;
   int myInt;
   unsigned int myUInt;
   float myFloat;
   double myDouble;
   f(myShort);
   f(myInt);
   f(myFloat);
   f(myUInt);
   f(myDouble);
}
```

Answers:

A) f(myShort)
B) f(myInt)
C) f(myFloat)
D) f(myUInt)
E) f(myDouble)

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1,5,10			
C++ Associate (CPA)	Chapter: 3	Section: 1,5,10	Question type: Multiple-blanks
Subject: Transferring data to functions			Question Number: 4

Question: The values of the following variables are a = ..., b = ..., c=..... .
They are.... memory leaks in the program.

```
void ModifyVariables(int a, int &b, int *c){
   a = b;
   b+=a;;
   c = new int(b);
   (*c)++;
}
int main(){
   int a=0,b=1, *c;
   c = new int(2);
   ModifyVariables(a,b,c);
   cout<<a<<b<<*c;
   delete c;
  }
```

Answers: [see question instructions above]

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1-5			
C++ Associate (CPA)	Chapter: 3	Section: 1-5	Question type: Single-choice
Subject: Passing data to functions			Question Number: 5

Question: What does the following code fragment in C++ do? (assumption: all #include and the rest of the code are correct)

```
void Pointer(int *p){
  (*p)++;
  Reference(*p);
  cout<<*p;
}
void Reference( int &p){
   p++;
   Value(p);
   cout<<p;
}
void Value (int p){
   p++;
   cout<<p;
}
int main(){
   int value =3;
   Pointer(&value);
   cout<<value;
}
```

Answers:

A) 6555
B) 6565
C) 5656
D) 6565

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1,2			
C++ Associate (CPA)	Chapter: 3	Section: 1,2	Question type: Single-choice
Subject: Passing data to functions			Question Number: 6

Question: What does the following code fragment in C++ display? (assumption: all #include and the rest of the code are correct)

```
int main(){
   int *index;
   int * vector;
   int i =128;
   vector = new int[5];
   index =vector;
   while (i){
      if (i%2 == 0)
         *index = i;
      i/=2;
      if ((vector + 4) == index )
         break;
      index++;
   }
   cout<<vector[i%5]<<endl;
   delete[] vector;
}
```

Answers:

A) 8
B) 16
C) 128
D) 64

ANSWER KEY

Correct answers:
Q1 - C

Explanation: C is correct because :
myInt1 = ++(*pnt1) + (*pnt1); // we first pre increment the value of pnt1 (since C++11 this behavior is not considered undefined) and then we add the values .
myInt2++; // sets the value of myInt2 to 4
myInt2 = (*pnt2)++ + (*pnt2); // pnt2 is a pointer to myInt2 (so when we dereference it returns 4 – the value of myInt2)

Correct answers:
Q2 - B

Explanation: B is correct because
SetElement function – sets the element from position “index” of the array “array” to value “value”: array[index] =value.
The first call sets the “point2[0]” to 0 and the second call sets ” point2[1]” to 1

Correct answers:
Q3 - C,D,E

Explanation:C,D,E :
C – the call “f(myFloat)” is ambiguous to the compiler(two functions match: “f(float)” and “f(float, int =0)”
D,E – the compiler you cannot convert “uint” to “int” and “double” to” float”

Correct answers:
Q4 - The values for the following variables are a = 0, b = 2, c = 2. They are 1 memory leaks

Explanation:
The variable “a” is transmitted by value (there is made a copy of the value of variable “a” so in the expression “a =b ” is modified the value of the local variable “a” (from the function)
The variable “b” is transmitted by reference and therefore his values is modified
Even the variable “c” is a pointer , the pointer value itself is transmitted by value, so when you want “c” to point to another memory zone actually the copy made to the “c” pointer points to a new memory zone.
There is one memory leak because you are allocating memory in the “ModifyVariables“ function that is never freed.

Correct answers: Q5 - A
Explanation: The “Value” function parameter is copied by value (so any changes made by the function to the variable are only local) The rest of the functions modify the content of the initial variable.
Correct answers: Q6 - A
Explanation: When we declare “new int[5]” the compiler allocates 5 consecutive blocks of memory of size int. “vector” points to the first element of the allocated memory block. “index++” goes to the next element from the block of memory (the next int) so “index+4” is the fifth element of the vector.

AUTHOR:	PRACHI PODDAR	PRACHI.PODDAR108@GMAIL.COM
AUTHOR'S BIO:	Prachi works at EdgeVerve, India as a product engineer (research and development). Her areas of interest are big data analysis, databases and conceptual programming in C, C++ & JAVA. Her hobbies are playing tennis and reading books.	

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1	Pointers: another kind of data in the "C++" language		
C++ Associate (CPA)	Chapter: 3	Section: 1	Question type: single-choice
Subject: Pointers in C++			Question Number: 1

Question: Which of the following statements is correct?

-

A) Base class pointer cannot point to derived class.
B) Derived class pointer cannot point to base class.
C) Pointer to derived class cannot be created.
D) Pointer to base class cannot be created.

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 8	Overloaded functions		
C++ Associate (CPA)	Chapter: 3	Section: 8	Question type: multiple-choice
Subject: Function Overloading			Question Number: 2
Question: Which of the following is correct about function overloading?			
-			
A) The types of arguments are different. B) The order of arguments is different. C) The number of arguments is different. D) The return type of the function is same.			

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 1	Pointers: another kind of data in the "C++" language		
C++ Associate (CPA)	Chapter: 3	Section: 1	Question type: single-choice
Subject: Pointers in C++			Question Number: 3
Question: What is the implicit pointer that is passed as the first argument for non-static member functions?			
A) 'self' pointer B) std::auto_ptr pointer C) 'Myself' pointer D) 'this' pointer			

Chapter: 3	Extending the expressive power: pointers, functions and memory		
Section: 7	Inline functions		
C++ Associate (CPA)	Chapter: 3	Section: 7	Question type: single-choice
Subject: Inline Functions			Question Number: 4
Question: Inline functions are invoked at the time of:			
A) Run time B) Compile time C) Depends on how it is invoked			

ANSWER KEY

Correct answers: Q1 – B
Explanation: no explanation
Correct answers: Q2 - A, B, C
Explanation: Return type has nothing to do with method/function overloading
Correct answers: Q3 – D
Explanation: no explanation
Correct answers: Q4 - B
Explanation: no explanation

AUTHOR:	VITALI KREMEZ	VKREMEZ@HOTMAIL.COM
AUTHOR'S BIO:	Becoming a programmer is deeply connected—the three year-long study of cybersecurity that students learn about on their first college day and do not stop thinking about until their last. It forces them to draw from all they have learned. It is my test of persevera1zce, creativity, and knowledge that appeared to be also, rather unexpectedly, the catalyst in my decision to study C++ programming. Vitali Kremez, CFE, CNDA, CEH, Sec+, Linux+, LPIC1, Suse CLA.	

Chapter: 3	Chapter 3: Extending the expressive power: pointers function and memory		
Section: 1.22	Another new operator		
C++ Certified Programmer Associate (CPA)	Chapter: 3	Section: 1	Question type: Multiple-choice
Subject: 3.1.22 Another new operator			Question Number: 1

Question: Fill in the following blanks to declare an array of integers that has 19 elements. You should assign the value of 50 to each element using the for loop.

```
#include <iostream>
using namespace std;

int main(){
int cpp_int[100];
for(int i =0; i<100; i++){
cout << "The size of the 'array' is " << sizeof(cpp_int) << " bytes";
return 0;
}
}
```

Answers:

A. 100;
B. 400;
C. 800;
D. 90

Chapter: 3	Chapter 3: Extending the expressive power: pointers, functions and memory		
Section: 3.2.2	Pointers vs. arrays		
C++ Certified Programmer Associate (CPA)	Chapter: 3	Section: 2	Question type: Multiple-choice
Subject: Pointers vs. arrays			Question Number: 2

Question: What is the output of the code below?

```
#include <iostream>
using namespace std;

int main() {
	int y[4] = { 77, 66, 55, 44 }, *ptr = y + 11;
	(*(ptr + 11))++;
	*ptr++;
	cout << y[1] << y[2];
	return 0;
			}
```

Answers:

A. 7766
B. 6655
C. 5544
D. 7755

ANSWER KEY

Correct answer: Q1 - C. 400
Explanation: N/A
Correct answer: Q2 - B. 6655
Explanation: N/A

AUTHOR:	SRAJAN RATTI	MSFREEDOM911@GMAIL.COM
AUTHOR'S BIO:	I am a Final year student of Amity University India. I am doing my Engineering in BTECH CSE.I love to learn new languages. My hobbies are Playing keyboard, making music ,animation.	

Chapter: [3]	Extending the expressive power: pointers, functions and memory		
Section:[1]	Pointers: another kind of data in the "C++" language		
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[1]	Question type: [Multiple-choice]
Subject: [References]			Question Number: [1]

Question: What would be the output of second cout ?

```
//assume size of integer to be 4 bytes
#include<iostream>
int main()
{

        int i = 5;
        int &r = i;
        std::cout << &r << "\n";
        r+=2;
        std::cout << &r << "\n";
    return 0;

}
```

Answers:

A. Output of both the cout would be same .
B. Value printed in second cout would be greater than value printed in first cout by 8.
C. Value printed in second cout would be lesser than value printed in first cout by 8.
D. Run Time or Compile Timer Error.

Chapter: [3]	Extending the expressive power: pointers, functions and memory		
Section:[1]	Pointers: another kind of data in the "C++" language		
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[1]	Question type: [Multiple-choice]
Subject: [Incrementing Pointers]			Question Number: [2]

Question: What would be the output of the following code ?

```
#include<iostream>
int main()
{

        int i = 5;
        int *j = &i;
        void *x=j;
        std::cout<<*x++;
    return 0;


}
```

Answers:

A. 6 .
B. Error--> void * unknown size .
C. 5.
D. Garbage Value.

Chapter: [3]	Extending the expressive power: pointers, functions and memory		
Section:[1-5]			
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[1-5]	Question type: [Multiple-choice]
Subject: [Pointers and functions]			Question Number: [3]
Question: What does the following function declaration means ?			
char *(*abc)(float *,float **);			

Answers:

A. abc is a   pointer to a char taking pointer to a float and pointer to a pointer to a float.
B. Error .
C. abc is a function taking pointer to a float and pointer to a pointer to a float and returning pointer to a pointer to a char.
D. abc is a pointer to a function taking pointer to a float and pointer to a pointer to a float as argument and returning a pointer to char.

Chapter: [3]	Extending the expressive power: pointers, functions and memory		
Section:[8]	Overloaded functions		
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[8]	Question type: [Multiple-choice]
Subject: [Overloaded functions]			Question Number: [4]

Question: What is the output of the following code ?

```
#include<iostream>
class XX{
int is;
public:
XX(int x){
is = x;
}
void display(){
std::cout << "XX";
}
};

class YY{
int is;
public:
YY(int x){
is = x;
}
void display(){
std::cout << "YY";
}
};

XX f(XX a){
a.display();
return a;}
YY f(YY b){
b.display();
return b;}

int main()
{f(5);
}
```

Answers:

A. XX.
B. YY.
C. XXYY.
D. YYXX.
E. Error.

Chapter: [3]	Extending the expressive power: pointers, functions and memory		
Section:[5]	Transferring data to and from functions		
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[5]	Question type: [Multiple-choice]
Subject: [Pointers to Functions]			Question Number: [5]

Question: What is the output of the code ?

```
#include<iostream>
int yO(int xx)
{
std::cout << xx;
return xx;
}

1.  int main()
2.  {
3.  int(*__foos$)(int);
4.  __foos$ = &yO;
5.  (*__foos$)(100);
6.  }
```

Answers:

A. Error at line 3.
B. Error at line 4 and 5.
C. 100.
D. Address of variable.

Chapter: [3]	Extending the expressive power: pointers, functions and memory		
Section:[4]	Declaring and defining functions		
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[4]	Question type: [Multiple-choice]
Subject: [Function syntax]			Question Number: [6]
Question: What is the syntax of strncat function ?			

//_Dest is Destination String and _Source is SourceString and num is number of //character to be copied

A. strncat(char *_Dest, const char *_Source, size_t num);
B. strncat(size_t num ,char *_Dest, const char *_Source);
C. strncat(const char *_Source,char *_Dest, size_t num);
D. strncat(size_t num ,const char *_Source,char *_Dest,);

Answers:

A. A is correct
B. B is correct.
C. C is correct.
D. D is correct.

Chapter: [3]	Advanced flow control and data aggregates		
Section:[2]	More types and when we use them		
C++ Certified Associate Programmer (CPA)	Chapter: [3]	Section:[2]	Question type[Multiple-choice]
Subject: [decltype]			Question Number: [7]

Question: What is the output of the following code?

```
#include <iostream>

int main() {

        char s = 'a';

        decltype((s)) bx = s;
        bx++;
        std::cout << "\n" << bx;
}
```

Answers:

A. a .
B. b.
C. c.
D. Error

ANSWER KEY

Correct answers: Q1 - A.
Explanation: Value of reference Cannot be changed.
Correct answers: Q2 - A.
Explanation: The size of the object pointed to is unkown.
Correct answers: Q3 - D.
Explanation: Use spiral Rule to answer the question. http://c-faq.com/decl/spiral.anderson.html
Correct answers: Q4 - E.
Explanation: - Clearly there is an ambiguity while calling function f(5)
Correct answers: Q5 - C.
Explanation: -
Correct answers: Q6 - A.
Explanation: -
Correct Answer: Q7 – C
Explanation: