Programming in C++: Assignment Week 2

September 20, 2017

**Question 1**

- Look at the code snippet below:

```cpp
const float * ch = &c ;
```

Which of the following statement is true for the variable ‘ch’?

- a. const-Pointer to non-const-Pointee
- b. non-const-Pointer to const-Pointee
- c. const-Pointer to const-Pointee
- d. non-const-Pointer to non-const-Pointee

**Answer:** b

**Explanation:** As per syntax

**Question 2**

- Look at the following code segment and decide which statement(s) is/are correct. Mark 1

```cpp
int main(){
    char m = 4;
    const char n = 5;
    const char * p = &n;
    char * const q = &m;
    // ...
    n = 6;  // stmt-1
    *p = 7; // stmt-2
    p = &m; // stmt-3
    *q = 8; // stmt-4
    ...
}
```

- a. stmt-1
- b. stmt-2
- c. stmt-3
- d. stmt-4

**Answer:** c), d)

**Explanation:** As per syntax
Question 3

- Identify the output of the following code. Mark 1

```cpp
#include<iostream>
using namespace std;
int main() {
    typedef union Complex {
        double re;
        double im;
    } Complex;

    const Complex c = {46} ;
    c.im = 59;
    cout << c.re;
    return 0;
}
```

a. 46 

b. Compilation Error: Cannot assign an integer value to a double variable 

c. Compilation Error: 'im' is a read only object 

d. 59 Answer: c 

Explanation: im is variable of the structure Complex, but it is defined as const, hence cannot be modified

Question 4

- Identify the correct statement(s). Mark 1

```cpp
#include <iostream>
#include <cmath>
using namespace std;
#define TWO 2
#define PI 4.0*atan(1.0)
int main() {
    int r = 10;
    double peri = TWO * PI * r;
    cout << "Perimeter = " << peri << endl;
    return 0;
}
```

a. Types of TWO is determinate 

b. TWO is a manifest constant 

c. Type of PI may be indeterminate 

d. PI look like variable

Answer: b) c), d)

Explanation: TWO and PI are manifest constants, hence types can be indeterminate and look like variables.
Question 5

- What will be the output of the following code? 

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

double increment(const double &prm) {
    return (prm + 1);
}

int main() {
    double x = 10, y;
    y = increment(x);
    cout << x+2 << " " << y;
    return 0;
}
```

a. 13 11  
b. 10 11  
c. 12 11  
d. 11 11  

**Answer:** c)  
**Explanation:** Const used to pass reference parameter prm to prevent from being modified. The value of prm is used only.

Question 6

- What will be the output of the following code? 

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

void func(int n1 = 14, int n2) {
    cout << n1 << " " << n2;
}

int main() {
    func(1);
    func(2.5, 4);
    return 0;
}
```

a. 1 14 2.5 4  
b. 14 1 2.5 4  
c. 14 1 2 4  
d. Compilation error: Default value missing for parameter 2 of func  

**Answer:** d)  
**Explanation:** `fun(1)` to work, Default values needs to specified in 2nd argument. Function resolution fails for `func(1)`
Question 7

- What will be the output of the following code?  
  
  ```cpp
  #include <iostream>
  using namespace std;
  int Add(int a, int b = 19) { return (a + b); }
  double Add(double c) {
    return (c + 1);
  }
  int main() {
    int x = 5, y = 4, z;
    z = Add(x, y);
    cout << z;
    double s = 9.5, u;
    u = Add(s);
    cout << " " << u << endl;
    return 0;
  }
  ```

  a. 9 28
  b. Add cannot be resolved (ambiguous)
  c. 9 28.5
  d. 9 10.5

  **Answer:** d)  
  **Explanation:** Two versions of function Add called as per resolution, with priority to the exact call

Question 8

- Which function prototype will match the function call `func(45.2, 65)`?

  ```cpp
  void func(int, int); // Proto 1
  void func(int, double, int = 6); // Proto 2
  void func(double, double, char = 'c'); // Proto 3
  void func(double, char = 'd', char = 'c'); // Proto 4
  ```

  a. Proto 1
  b. Proto 2
  c. Proto 3
  d. Proto 4

  **Answer:** a), b), c), d)  
  **Explanation:** Proto 1 allowed, as 45.2(1st parameter) is downcast to integer. Proto 2 allowed, as default value will be used for third parameter. Proto 3 allowed, default value and type will be used for third parameter. Proto 4 fills for mismatch in 2nd parameter
Question 9

• What will be the output of the following code? 

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

int& Ref_func( int param) {
    return (++param);
}

int main() {
    int x = 10, y = 15, z = 14;
    y = Ref_func(x);
    cout << x << " " << y;
    Ref_func(x) = z;
    cout << x << " " << y;
    return 0;
}
```

a. 10 11 10 11
b. 11 15 11 15
c. Compilation Error: invalid function call

d. Compilation Error: invalid assignment of pointer to non-pointer Answer: a)

Explanation: reference to local parameter returned

Question 10

• Fill up the blanks to get the desired output according to the test cases.

```cpp
#include <iostream>
#include <cstring>
#include <cstdlib>
using namespace std;

typedef struct _String { char *str; } String;

String s;
s.str = (char *) malloc(strlen(s1.str) + strlen(s2.str) + 1);
strcpy(s.str, s1.str);
strcat(s.str, s2.str);
return s;

int main() {
    String s1, s2, s3;
    s1.str = strdup("I");
    s2.str = strdup(" love Travelling ");
    s3 = s1 + s2;
    cout << s3.str << endl;
    return 0;
}
```

5
a. String operator+(const String& s1, const String& s2)
b. String + operator(const String& s1, const String& s2)
c. String +(const String& s1, const String& s2)
d. string operator+(const String s1, const String& s2)

Answer: a)
Explanation: As per syntax, Overloading operator + for String structure. Reference parameters passed as const to prevent modification.

Question 11

- What will be the output of the following code?  
  
#include <iostream>
using namespace std;

inline int SQR(int x) { return x * x; }

int main() {
    int a, b, c;
    a = 10, b = 14;
    b = SQR(a);
    cout << b << endl;
    c = SQR(++a);
    cout << c << endl;
    return 0;
}

a. 100 121
b. Compilation Error: invalid function definition
c. 100 132
d. Compilation Error: invalid function parameter

Answer: a)
Explanation: inline function, as per syntax

Question 12

- What will be the output of the following code?

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

int main() {
    int d;
    int *p = (int *) operator new(sizeof(int));

    d = 5;
    *p = d;
    cout << +++p + d++;

    return 0;
}
a. 10
b. 11
c. 12
d. Compilation Error: pointer not deleted after allocation with new **Answer:** b)
   **Explanation:** As per syntax of pointers