AMCAT Procedure functions and scope Sample Questions

Question 1

A function cannot be defined inside another function

A. True

B. False

Answer: Option A

Explanation: A function cannot be defined inside the another function, but a function can be called inside a another function.

Question 2

Functions cannot return more than one value at a time

A. True

B. False

Answer: Option A

Explanation: True, A function cannot return more than one value at a time. Because after returning a value the control is given back to calling function.

Question 3

How many times the program will print "Alpingi" ?

```
#include<stdio.h>
int main()
{
   printf("Alpingi");
   main();
   return 0;
}
```

A. Infinite times

B. 32767 times

C. 65535 times

D. Till stack overflows

Answer: Option **D**

Explanation: A call stack or function stack is used for several related purposes, but the main reason for having one is to keep track of the point to which each active subroutine should return control when it finishes executing. A stack overflow occurs when too much memory is used on the call stack.
Here function main() is called repeatedly and its return address is stored in the stack. After stack memory is full. It shows stack overflow error.

Question 4

The default parameter passing mechanism is

A. call by value

B. call by reference

C. call by value result
D**.** None of these.
Answer: Option **A**

Question 5
Which of the following function calculates the square of ‘x’ in C?
A**.** sqr(x)
B. power(x, 2)
C. power(2, x)
D. pow(x, 2)
E. pow(2, x)
Answer: Option **B**

Question 6
What will be the output of the program?

```
#include<stdio.h>
int i;
int fun();
int main()
{
   while(i)
   {
     fun();
     main();
    }
   printf("Hello\n");
   return 0;
}
int fun()
{
  printf("Hi");
}
```

A. Hello
B. Hi Hello
C.No output
D.Infinite loop

Answer: Option **A**
Explanation: Step 1: int i; The variable i is declared as an integer type.
Step 2: int fun(); This prototype tells the compiler that the function fun() does not accept any arguments and it returns an integer value.
Step 3: while(i) The value of i is not initialized so this while condition is failed. So, it does not execute the while block.

Step 4: printf("Hello\n"); It prints "Hello".
Hence the output of the program is "Hello".

Question 7

What will be the output of the program?

```
#include<stdio.h>
int main()
{
   int fun(int);
   int i = fun(10);
   printf("%d\n", --i);
   return 0;
}
int fun(int i)
{
  return (i++);
}
```

A. 9
B. 10
C. 11
D. 8

Answer: Option **A**

Explanation: Step 1: int fun(int); Here we declare the prototype of the function fun().
Step 2: int i = fun(10); The variable i is declared as an integer type and the result of the fun(10) will be stored in the variable i.
Step 3: int fun(int i){ return (i++); } Inside the fun() we are returning a value return(i++). It returns 10. because i++ is the post-increement operator.
Step 4: Then the control back to the main function and the value 10 is assigned to variable i.
Step 5: printf("%d\n", --i); Here --i denoted pre-increement. Hence it prints the value 9.

Question 8

What will be the output of the program?

```
#include<stdio.h>
int main()
{
   int i=1;
   if(!i)
      printf("IndiaBIX,");
  else
  {
      i=0;
      printf("C-Program");
```

```
      main();
   }
  return 0;
}
```

A. prints "IndiaBIX, C-Program" infinitely
B. prints "C-Program" infinetly
C. prints "C-Program, IndiaBIX" infinitely
D. Error: main() should not inside else statement

Answer: Option B

Explanation: Step 1: int i=1; The variable i is declared as an integer type and initialized to 1(one).
Step 2: if(!i) Here the !(NOT) operator reverts the i value 1 to 0. Hence the if(0) condition fails. So it goes to else part.
Step 3: else { i=0; In the else part variable i is assigned to value 0(zero).
Step 4: printf("C-Program"); It prints the "C-program".
Step 5: main(); Here we are calling the main() function.
After calling the function, the program repeats from step 1 to step 5 infinitely.
Hence it prints "C-Program" infinitely.

Question 9

Point out the error in the program

```
f(int a, int b)
{
   int a;
   a = 20;
   return a;
}
```

A. Missing parenthesis in return statement
B. The function should be defined as int f(int a, int b)
C. Redeclaration of a
D. None of above

Answer: Option C

Explanation: f(int a, int b) The variable a is declared in the function argument statement.
int a; Here again we are declaring the variable a. Hence it shows the error "Redeclaration of a"

Question 10

Point out the error in the program

```
#include<stdio.h>
int main()
{
   int a=10;
   void f();
   a = f();
```

```
    printf("%d\n", a);
    return 0;
 }
void f()
{
   printf("Hi");
}
```

A. Error: Not allowed assignment

B. Error: Doesn't print anything

C. No error

D. None of above

Answer: Option **A**

Explanation: The function void f() is not visible to the compiler while going through main() function. So we have to declare this prototype void f(); before to main() function. This kind of error will not occur in modern compilers.