Visit QB365-SLMS - https://www.qb365.in/slms/ Complete Questions and Answers

Answers Key:

Long Answer Type Questions

1. books=[ ]
   defAddnew (books,name):
   books.append (name)
   print("bookl:",name,"inserted")
   def remove(books):
   if books==[]:
   print("stack is empty")
   else:
   print("book:",books.pop ( ), "deleted ")
2. I. L=[("Laptop", 90000),("Mobile", 30000).("Pen", 50). ("Headphones". 1500)]
   product = [ ]
   defPush_element(L):
   for i in L:
   if i[1] > 50:
   product.append(i)
   print(product)
   II. defPop_element (product):
   while product:
   print(product.pop())
   else:
   print("Stack Empty")
3. def isempty(stk):
   if stk == [ ] :
   return True
   else:
   return False
   def pushbook(stk, item):
   stk.append(item)
   def displaybook(stk):
   if isempty(stk):
   print("Stack Empty")
   else:
   print("Book No... Book Name")
   for a in range(len(stk)-1, -1, -1):
   print(stk[a])
   stk = [ ]
   while True:
   print("\nBook Stack Menu")
   print("1. Add a book")
   print("2. Display list")
   print("3. Exit")
   ch = int(input("Enter your choice 1-3: "))
   if ch == 1:
   bno = input("Enter book no: ")
   bname = input("Enter book name: ")
   item = [bno, bname]
   pushbook(stk, item)
   input("Book added. Press Enter to continue...")
   elif ch == 2:
   displaybook(stk)
   input("Press Enter to continue...")
   elif ch == 3:
   break
   else:
   print("Invalid choice!")
   input("Press Enter to continue...")
4. (I) def push_num(NumStack, new_num)
   if new_numleft==0:
   NumStack.append(new_num)

(II) def pop_num(NumStack):
if not NumStack:
print("Underflow")
else:
return NumStack.pop( )
(III) def peep(NumStack):
if not NumStack:
print("None")
else:
print(NumStack[-1])

Short Answer Type Questions

5. 1. numberList starts as [10, 20, 30].
2. numberList.append(40) adds 40 to the list, so numberList becomes [10, 20, 30, 40].
3. numberList.pop() removes and returns the last item (40), so result 0+40=40.
4. numberList.pop() removes and returns the last item (30), so result = 40+30= 70.
5. The final print statement outputs Result= 70.
(b) The list answer starts as ['T', 'A', 'M'].
The pop() method removes the last element from the list and appends it to output.
The order of pops is:
1. 'M' is popped first, then added to output.
2. 'A' is popped, added to output.
3. "T" is popped, added to output.
Thus, the result is 'MAT'.
6. def reverse_string(s):
stack = [ ]
# Push all characters of the string onto the stack
for char in s:
stack.append(char)
# Pop characters from the stack to get the revers string
reversed_str = ""
while stack:
reversed_str += stack.pop( )
return reversed_str
input_str "hello"
print("Reversed string:",
reverse_string(input_str))
7. **# Function to create a Stack for odd numbers**
def store_odd_numbers ( ):
stack = [ ]
n = int(input("Enter the number of elements: "))
 # Input numbers and push only odd numbers to stack
for _ in range(n):
num = int(input("Enter a number: "))
if num % 2 ! = 0:
stack.append(num)
 # Display stack content and find the largest odd number
print("Stack contents (Odd numbers):",stack)
 # Finding the largest odd number
largest_odd=None
while stack :
current stack.pop( )
if largest_odd is None or current >largest_odd:
largest_odd = current
print("Largest odd number in the Stack:", largest_odd)
 # Call the function
store_odd_numbers ( )
8. The stack contents will be as follows after the operations of stack
(i) STACK : A, C, D, F
(ii) STACK : A, C, D
(K is deleted) (F is deleted)
(iii) STACK: A, C, D, L
(iv) STACK: A, C, D, L, P
(L is inserted) (P is inserted)
(v) STACK: A, C, D, L
(vi) STACK: A, C, D, L, R
(P is deleted) (R is inserted)
(vii) STACK: A, C, D, L, R, S

QB 365 TM Study Material
QUESTION BANK 365

(S is inserted)
(viii) STACK: A, C, D, L, R
(S is deleted)

9. BooksStack = [ ]
def push_book(stack, new_book):
stack. append(new_book)
def peep(stack):
if stack:
print(stack[-1])
else:
print("None")
# Operations
push_book (BooksStack, ["Python Basics", "John Smith", 2021])
push_book (BooksStack, ["Data Structures","Alice Brown", 2019])
print("Topmost Book:")
peep(BooksStack)

10. NumStack = [10, 20, 30.40]
def push(stack, value):
stack.append(value)
def pop(stack):
if stack:
print(stack.pop( ))
else:
print("Stack Empty")
# Operations
pop(NumStack)
push(NumStack, 50)
print(NumStack)

11. (A) (I) def push_book (BooksStack, new_book):
BooksStack.append(new_book)
Here the stack is implemented using a stack.
Hence the append() function is used to add the new element at the top of the stack.
(II) def pop_book (BooksStack):
if not BooksStack:
print("Underflow")
else:
return BooksStack.pop( )
Here the stack is implemented using a stack.
The pop( ) function removes and displays the last element of a list. Hence the pop( ) function is used here to take out the topmost element of the stack
(III) def peep (BooksStack):
if not BooksStack:
print("None")
else
print(BookStack[-1])
Since the topmost element has to be displayed without removing, the pop() function cannot be used here. Instead the BookStack[-1] statement prints the last element of the list that has index -1.

Or

(B) n=int(input("Enter an integer: "))
s =[ ] #stack
f = 2
while n>1:
if n%f ==0:
s.append(f)
n //=f
else:
f +=1
while s:
print(s.pop().end=' ')
To find the prime factorization the number is repeatedly divided to find the clear division, starting from f = 2 When no longer divisible by 2 it is divided by the next prime factor and so on. All such factors are pushed into the stack. Finally the stack contents are popped.

12. **# Define the stack and input list**
Nums [1234, 12345, 67890. 12. 567890, 123]
BigNums = [ ]
(i) def PushBig( ):
for N in Nums:
if N >= 10000:
 # Check if the number is 10000 or greater

QB365™ Study Material
QUESTION BANK 365

```
BigNums.append(N)
(ii) def PopBig( ) :
while BigNums :
print(BigNums.pop( ) )
print( "Stack Empty" )
# Test the functions
PushBig( )
PopBig( )
```

13.
```
travel = [ ]
(i) def Push_element(NList):
for i in NList:
if i[1]!="India" and 1[2]<3500:
travel . append([i[0],i[1]])
(ii) def Pop_element( ) :
while len(travel) :
print (travel . pop( ) )
else :
print("Stack Empty")
```

14.
```
(a) L = [['Siddharth', 'Delux'], ['Rahul". 'Standard']. ['Jerry'. 'Delux']]
Hotel = [ ]
(i) def Push_Cust( ):
for cust in L:
if cust[1]=='Delux':
Hotel.append(cust [0])
(ii) def Pop_Cust( ):
while len(Hotel)>0:
print(Hotel .pop( ) )
print("Underflow")
Push_Cust( )
Pop_Cust( )
```

Or

```
(b) Vehicle-{"Santro" : "Hyundai". "Nexon": "TATA". "Safari" : "TaTa"}
stack = [ ]
def Push (V):
for i in V:
if V[1].upper()== "TATA":
stack.append(i)
print(stack)
Push(Vehicle)
```

15.
```
(a) (i) def manage_homework(names) :
HWStack [ ]
for name in names :
HWStack.append(name)
(ii) while HWStack:
print(HWStack.pop( ) )
print("No more submissions left!")
# Example
manage_homework ( ["Amit", "Neha", "Karan"])
```

Or

```
(b) (i) FileStack = [ ]
def push_file(FileStack, filename):
FileStack.append(filename)
(ii) def pop_file(FileStack):
if not FileStack:
print("No files left to close!")
else:
print("Closed", FileStack.pop( ) )
# Example
push_file(FileStack, "notes.txt" )
push_file(FileStack, "project.py" )
pop_file (FileStack)
pop_file (FileStack)
pop_file (FileStack)
```

16.
```
(a) (i) def push_emp(Stack. EmpDict):
for name in EmpDict:
if EmpDict[name]> 50000
Stack append(name)
```

QB 365 Study Material
QUESTION BANK 365

(ii) def pop_emp(Stack)
if not Stack:
print("Stack Underflow")
return None
return Stack .pop( )
(iii) def display_emp(Stack):
if not Stack :
print("Empty Stack")
else :
print("Stack contents". Stack)

Or

(b) (i) def push_even (Stack. NumList):
for num in NumList:
if num % 2 == 0:
Stack.append(num)
(ii) def pop_num(Stack):
if not Stack :
print ("Stack Underflow")
return None
return Stack.pop( )
(iii) def display_stack(Stack) :
if not Stack :
print("Empty Stack")
else :
print("Stack contents:". Stack)

17. (A) (i)

```
R= {"OM":76, "JAI" : 45, "BOB" : 89, "ALI" : 65, "ANU" : 90, "TOM" : 82}
def PUSH (S,N):
S.append (N)
(ii) def POP(S):
if S! = [ ] :
return S.pop( )
else :
return None
ST = [ ]
for k in R:
if R[k]>=75:
PUSH(ST ,k)
while True:
if ST! = [ ] :
print(POP(ST), end=" ")
else :
break
```

Or

```
(B) palinstack=[ ]
def pushcity(n) :
palinstack.append(n)
ans='y'
while ans=='y':
n=input ("Enter a city:")
if n==n [: :-1]:
pushcity(n)
ans-input("Add more (y/n)")
```

18. (a) def enterstack(slst) :
if slst[3]=="A" :
Admissionstack.append(slst)
(b) def getstack ( ) :
if Admissionstack == [ ] :
print("No students")
else:
Admissionstack.pop( )
(c) def peepstack ( ):
if Admissionstack==[ ]:
print("No students")
else:
p= len(Admissionstack)-1
while p>=0 :

QB 365 Study Material
QUESTION BANK 365

```
print(Admissionstack[p])
p-=1
```

Or

(i) A stack allows the following operations:

Push( ) This function pushes an element into the stack keeping the LIFO principle, which means the element is pushed to the top of the stack.

Pop( ) This function takes out the topmost element from the stack and displays it.

Traversal ( ) This function displays each of the stack contents from top index to 0 with the LIFO principle. It does not remove the elements.

(ii)
```
stack = [ ]
def push2_5(n) :
if n%10==2 or n%10-5 :
stack.append(n)
```

19. (a) (i)
```
def push_Clr(ClrStack, new_Clr):
ClrStack.append (new_Clr)
```
(ii)
```
def pop_Clr(C)rStack) :
if is Empty (ClrStack) :
print ("Underflow")
return None
return ClrStack.pop ( )
```
(iii)
```
def isEmpty(ClrStack) :
return len(ClrStack) == 0
```

Or

(b) (i)
```
def push_trail (N, myStack) :
p = [- 5 : ] # This slice will be an empty list
myStack.extend(p)
```
(ii)
```
def pop_one(myStack) :
if len (myStack) == 0 :
print("Stack underflow")
return None
else:
item = myStack.pop ( )
return item
```
(iii)
```
def display_all(myStack) :
if len(myStack) == 0 :
print("Stack is Empty")
else
top = len(myStack) - 1
print("top ->", top)
for p in range(top 1.-1.-1):
print(myStack[p])
```

QB 365 Study Material
QUESTION BANK 365

20.
```
def PUSH (S,N) :
S . append (N)
def POP(S) :
if S ! = [ ] :
return S.pop ( )
else :
return None
ST = [ ]
fork in N :
if k%2==0 :
PUSH (ST , k)
while True :
if ST! = [ ] :
print(POP(ST), end=" ")
else:
break
```

21. Python function PUSH(Arr) that pushes numbers divisible by 5 into a stack (list) and displays the stack or an error message:
```
def PUSH(Arr) :
stack = [ ]
# Initialize an empty stack
for num in Arr :
if num % 50 == :
# Check if the number is divisible by 5
stack.append(num)
# Push the number into the stack
if stack :
# If the stack contains at least one element
print ("Stack:", stack)
```

else:
print("Error: No numbers divisible by 5.")
Example:
PUSH([1, 5, 10, 7, 15])
Output :
Stack: [5, 10, 15]
If there are no numbers divisible by 5 in the list:
PUSH ([1, 2, 3, 7])
Output:
Error: No numbers divisible by 5.

22. (a) # Function to push fruit onto stack
if Price < 100
def push_fruit(FruitStack, Fruit):
if Fruit['Price'] < 100:
FruitStack.append(Fruit)
else:
print("Fruit not added
(Price >= 100)")
# Function to pop topmost fruit
def pop_fruit(FruitStack):
if len(FruitStack) == 0:
print("UNDERFLOW")
return None
else:
return FruitStack.pop()
Function to display stack from top
to bottom
def display(FruitStack):
if len(FruitStack) == 0:
print("EMPTY STACK")
else:
print("Stack elements (Top to
Bottom):")
for i in range(len
(FruitStack)-1, -1, -1):
print(FruitStack[i])
# .....- Example Usage
FruitStack = [ ]
f1 = |'Name': 'Apple', 'Origin':
'France', 'Price': 120, 'Expiry':
'12-08-2025')
f2 = {'Name': 'Banana', 'Origin':
'India'. 'Price': 40. 'Expiry':
'10-04-2025' |
push_fruit(FruitStack, f1)
# Won't be added
push_fruit(FruitStack. f2)
# Will be added
display(FruitStack)
print("Popped:",
pop_fruit(FruitStack))
display(FruitStack)
(b) # Initialize stack
stack = [ ]
# Input 10 integers
for i in range(10):
num = int(input("Enter number: "))
# Check if number is 3-digit and
even
if num >= 100 and num <= 999 and
num % 2 == 0:
stack.append(num)
# Pop and display elements
print("Output:")
while len(stack) > 0:
print(stack.pop(), end=" ")

QB 365 Study Material
QUESTION BANK 365

Case Study Questions

23. (i) (C) As it is list within a list.
(ii) (C) Both are correct as L[4][1] and L[-2] [-2] access the same value from the list as 8.91.
(iii) (A) As the list inside another list is taken as one element, thus there are 6 total items in the list L.
(iv) (B) As both len(L[2] and len(L[3]) represents same length so the output on the screen will be true.
(v) (A) The output of the print command will be [3.45, 'Amar'] as the indexing corresponds to this items in the list.

24. (i) (B) The Statement 1 used under the push() operation. The Push operation of stack is used to insert element in the stack.
(ii) (D) The alternate element from the country list C will be 2 because the list starts with 0.
(iii) (C) The stack will be considered as empty, if all the elements are popped up from stack and its length has become 0.
(iv) (B) Statement 4 is a part of pop function. If condition is used to check whether stack is empty or not. If not then code will execute else part and pop element from stack.
(v) (B) The Statement 5 will call pop function, which is already declared as Code 2 and taking a parameter country list

QB 365 Study Material
QUESTION BANK 365