# مقرر برمجة 1

**الجلسة العملية الخامسة**
**الفصل الصيفي-2023/2022**

- موضوع الجلسة:
  - نداء تابع على توابع أخرى
  - النداء على التابع بالقيمة وبالمرجع .
  - القيم الافتراضية
  - التحميل الزائد

```
#include <iostream>
using namespace std;
int add(int num1,int num2)
{
   return num1+num2;
}

int main()
{  int n1 = 97,n2 =34, n3=45,n4=90, n5=65;

int result =add(add(add(add(n1,n2),n3),n4),n5);

cout<<result;
return 0;
}
```

التمرين الأول

عرف تابع يجمع قيمتين صحيحتين واستخدمه في برنامج رئيسي من أجل ايجاد قيمة مجموع خمسة أعداد

الغاية من التمرين أن يتعلم الطالب أن يستخدم التابع الذي يعيد قيمة كمضمن أثناء النداء على التابع

$$result = add(add(add(\underbrace{add(n1,n2)}_{r1},n3),n4),n5);$$
$$\underbrace{\qquad\qquad\qquad\qquad}_{r2}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{r3}$$

**التمرين الأول: تابع يعيد قيمة وينادي على توابع أخرى و يأخذ ثلاثة وسطاء**

عرف تابعاً يقوم بدور الالة الحاسبة
يأخذ هذا التابع ثلاثة بارامترات الاول و الثاني عددين صحيحين و الثالث من النوع المحرفي char يمثل العملية الرياضية.
ينفذ التابع عمليات الجمع و الضرب و الطرح على العددين الصحيحين.
بحيث تتم العمليات الحسابية الثلاث باستخدام توابع معرفة خارج تابع الآلة الحاسبة .
ثم استخدم تابع الالة الحاسبة في برنامج رئيسي.

```
#include <iostream>
using namespace std;
int add(int , int );
int sub(int , int );
int mul(int , int );

int calculator(int num1, int num2, char operation)
{   // calling add function within calculator function
   if (operation == '+')  { return add(num1, num2); }
// calling sub function within calculator function
 if (operation == '-')    {   return sub(num1, num2 }
// calling mul function within calculator function
 if (operation == '*')
     { return mul(num1, num2);}
}
// function for adding two numbersint
add(int num1, int num2)      { return (num1 + num2);}
// function for subtracting numbers
int sub(int num1, int num2)  {    return (num1 - num2);}
// function for multiplying two numbers
int mul(int num1, int num2) {    return (num1 * num2);}
```

```
int main(){
int num1 = 10, num2 = 5;
// variable  giving options for different  calculation(add, sub, mul)
char op;      op = '*' ; // Assuming that user inputs ‘*’
 cout << calculator(num1, num2, op);
 return 0;}
```

## التمرين الأول: تابع يعيد قيمة+ ويأخذ وسيطين

عرف تابعاً يرد المتوسط الحسابي لقيم عناصر مصفوفة أحادية البعد عناصرها أعداد صحيحة، واستخدمه في برنامج رئيسي يعرف مصفوفة من خمسة عناصر تعطى قيم ابتدائية {22, 11, 33, 44, 55} .

```
#include <iostream>
using namespace std;
// التصريح عن التابع
double average(int arr[], int size)
 {int sum = 0 ;
  for (int i = 0; i < size; ++i)
    { sum += arr[i]; }
  return double(sum)/size;}
int main ()
{int a[5] = {1000, 2, 3, 17, 50};//  التصريح عن مصفوفة واسناد القيم
 cout << "Average value is: " << average( a,5)<< endl; // مناداة التابع
 return 0;}
```

**التمرين الثاني: تابع لا يعيد شيء + يأخذ ثلاثة وسطاء الاخير التمرير بالمرجع**

عدل على التابع السابق ليكون الخرج من نمط void
ووضح التعديلات اللازمة في البرنامج الرئيسي.

```
#include <iostream>
using namespace std;
void  average(int arr[], int size, double & ave)
 { int sum = 0 ;
   for (int i = 0; i < size; ++i)
        {   sum += arr[i];  }
   ave = double(sum) / size;}
int main ()
{ int a[5] = {1000, 2, 3, 17, 50};
  double result;
average( a, 5 , result ) ; //call
  cout << "ave="<<result;
  return 0;}
```

**تمرين إضافي: تابع لا يعيد قيمة + يأخذ ثلاثة وسطاء بحيث يتم تمرير الوسطاء الثاني و الثالث بالمرجع**

```
#include <iostream>
#include <cmath>
using namespace std;
void circule(double r, double & circumference ,
double & area){
// M_PI = 3.1415
   circumference = 2 * M_PI * r;
   area = M_PI * r * r;}
int main()
{
   double r, result1 , result2;
   r = 3;
   circule(r, result1, result2);
   cout << result1 << "   " << result2 << endl;
}
```

اكتب تابع من نوع void يقوم بحساب محيط ومساحة دائرة معلوم نصف قطرها،
يمرر له 3 بارامترات
البارامتر الأول هو بارامتر دخل وهو عبارة عن نصف القطر .
البارامتران الثاني والثالث عبارة عن المتحولين الذين ستخزن ضمنهما قيمة المحيط والمساحة.

**ملاحظة يعتمد هذا التمرين على تمرير البارامترات للتابع بالمرجع , لماذا؟**

```
#include <iostream>
using namespace std;
int AreaCube(int length, int width = 25, int height = 1){
return (length * width * height); }
int main()
{
int length = 100, width = 50 ,height = 2,new_height=7,area;
area = AreaCube(length, width, height);
cout << "First area equals: " << area << "\n";
area = AreaCube(90, 40, 5);
cout << “Second area equals: " << area << "\n";
area = AreaCube(length, 30, 4);
cout << "First area equals: " << area << "\n";
area = AreaCube(length, width);
cout << "Second time area equals: " << area << "\n";
area = AreaCube(length);
cout << "Third time area equals: " << area << "\n";
area = AreaCube(new_height);
cout << "Forth time area equals: " << area << "\n";
area = AreaCube(height);
cout << "Forth time area equals: " << area << "\n";
return 0;
}
```

جامعة المنارة
MANARA UNIVERSITY

## التمرين الثالث: تابع ذو قيم افتراضية

يسمح استعمال التابع ذو القيم الافتراضية

أوجد خرج البرنامج التالي مع تتبع الخطوات

## تمرين إضافي

اكتب تابع يقوم بطلب فاتورة عميل ويطبق عليها حسم قدرة يحدده الكاشير، وفي حال لم يحدد الكاشير قيمة الحسم عندها يكون الحسم الافتراضي هو 5%، ثم يعيد المبلغ الواجب دفعه

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

double fn(double bill, double discount = 0.05){
   return bill - bill * discount;
}

int main()
{
   double bill = 5000;
   cout << fn(bill) << endl;
   cout << fn(bill, 0.1) << endl;
}
```

```
#include <iostream>
using namespace std;
// التصريح عن التابع
int sum(int a, int b)
 {int sum = 0 ;
  sum =a+b;
 return sum;}
int sum(int a, int b,int c)
 {int sum = 0 ;
  sum =a+b+c;
  return sum;}
double sum(double a, double b)
 {double sum = 0 ;
  sum =a+b;
return sum;}
int main ()
{
cout << “sum  is: " << sum( 8,5)<< endl;
cout << “sum  is: " << sum( 8.8,7.6)<< endl;
cout << “sum  is: " << sum( 8,9,5)<< endl;

return 0;}
```

**التمرين الرابع: التحميل الزائد للتوابع function overload_**

**يعني أن تابعا واحدا يؤدي العديد من المهام ضمن نطاق نفس البرنامج الرئيسي .**
**احتفظ بهذه المعلومة المفيدة ان التحميل الزائد للتوابع ما هو الا الانعكاس الطبيعي لمبدأ تعددية الأشكال (Polymorphism) و الذي يعني أن نفس الكينونة(entity) تتصرف بشكل مختلف وفقا للسيناريوهات المختلفة.**

اكتب تابع sumيقوم بحساب مجموع:

- عددين صحيحين
- عددين حقيقين
- ثلاثة أعداد صحيحة

## التمرين السادس

اكتب

- تابع يسمح بتمرير وسيط من النمط الصيحيح ولا يعيد شيء ويقوم باضافة درجة واحدة الى هذا الوسيط اذا كانت القيمة تساوي 59 و درجتين اذا كانت القيمة تساوي 58.
- برنامج يقوم بتعريف مصفوفة ثنائية لعلامات طالبين لكل طالب اربع علامات حيث يتم تهيئة المصفوفة بالقيم المطلوبة واستخدم تابع السابق لتطبيق المساعدة الجامعية
- اطبع المصفوفة الناتجة

```
#include <iostream>
using namespace std;
void update(int& num)
{
   if (num == 58)
      num += 2;
   if (num == 59)
      num++;
}
int main()
{int marks[2][4] = {{50, 59, 74, 63}, {90, 84, 92, 82}};
   for (int t = 0; t < 2; t++)
      for (int k = 0; k < 2; k++)
         update(marks[t][k]);
   for (int t = 0; t < 2; t++)
   {for (int k = 0; k < 2; k++)
         cout << marks[t][k] << "\t";
      cout << endl; }
     return 0; }
```

## التمرين الخامس

عرف تابعاً للبحث عن قيمة ما ضمن مصفوفة أحادية البعد من الأعداد الصحيحة، حيث

- يرد 1- اذا لم تكن القيمة موجودة

أو يرد فهرس القيمة بحال وجودها

على أن يرد هذه القيم باستخدام المرجع.

- ثم استخدمه في برنامج رئيسي.

```
#include <iostream>
using namespace std;
void  printind (int ARR[],int size1,int value,int & ind );
int main()
{ int n,val,ans;
  int a[100];
  cout<<"n="; cin>>n;
  for (int i=0; i<n; i++)
     {cin>> a[i];}
  cout<<"val=";   cin>>val;
  printind (a, n,val,ans);
  cout<<ans<< endl;;
  return 0;}
void  printind (int ARR[],int size1,int value,int &ind)
{ ind = -1;
 for (int i = 0;i<size1;i++)
      {if (ARR[i]==value)
          {ind = i ;}}}
```

جَامعة
المَنارة
MANARA UNIVERSITY

# انتهت الجلسة