# САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

**ИНСТИТУТ КОМПЬЮТЕРНЫХ НАУК И КИБЕРБЕЗОПАСНОСТИ**

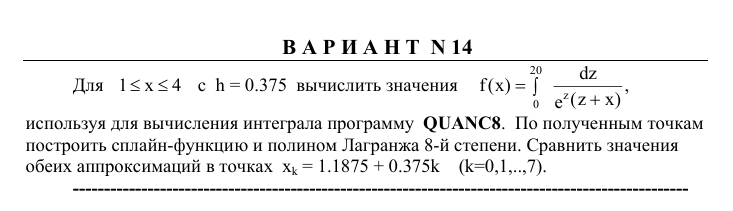
# Направление подготовки: 09.03.04 «Программная инженерия» б5130904/30022

**Задание №1, вариант 14.**

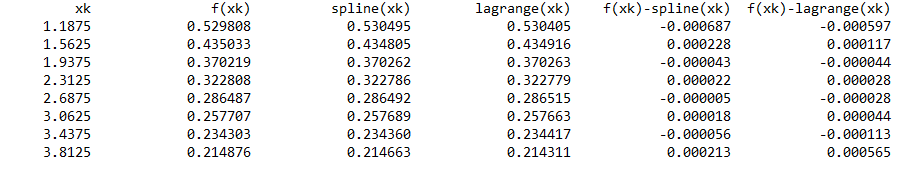
Выполнил студент: Лютов Александр Владимирович, группа 30022.

Преподаватель: Воскобойников Сергей Петрович.

# Задание



**Результат работы программы**



# Выводы

**Разница между функцией и сплайном:**

* Максимальная разница: **0.000213** (при x=3.8125*x*=3.8125)
* Минимальная разница: **-0.000687** (при x=1.1875*x*=1.1875)
* Средняя разница: близка к нулю, но есть небольшие отклонения.

**Разница между функцией и полиномом Лагранжа:**

* Максимальная разница: **0.000565** (при x=3.8125*x*=3.8125)
* Минимальная разница: **-0.000597** (при x=1.1875*x*=1.1875)
* Средняя разница: также близка к нулю, но отклонения больше, чем у сплайна.

Из представленных данных видно, что **сплайн** обеспечивает более точную аппроксимацию функции, чем полином Лагранжа. Разница между функцией и сплайном в большинстве точек меньше, чем разница между функцией и полиномом Лагранжа. Это особенно заметно в точках x=3.4375*x*=3.4375 и x=3.8125*x*=3.8125, где разница для полинома Лагранжа достигает **-0.000113** и **0.000565** соответственно, в то время как для сплайна она составляет **-0.000056** и **0.000213**.

Таким образом, **сплайн** в данном случае лучше аппроксимирует функцию, чем полином Лагранжа.

# Код программы

# main.f90

program main

use integral\_func\_mod

implicit none

interface

subroutine SPLINE(N, X, Y, B, C, D)

integer, intent(in) :: N ! Число заданных точек или узлов

real, intent(in) :: X(N) ! Абсциссы узлов в строго возрастающем порядке

real, intent(in) :: Y(N) ! Ординаты узлов

real, intent(out) :: B(N), C(N), D(N) ! Массивы определенных выше коэффициентов сплайна

end subroutine SPLINE

function SEVAL(N, Xi, X, Y, B, C, D) result(seval\_value)

integer, intent(in) :: N

real, intent(in) :: Xi, X(N), Y(N), B(N), C(N), D(N)

real :: seval\_value

end function SEVAL

end interface

external quanc8

real :: a, b, relerr, abserr, res, errest, flag

integer :: nofun

real, allocatable :: x\_values(:), f\_values(:)

real, allocatable :: b\_coef(:), c\_coef(:), d\_coef(:)

integer :: i, k, x\_n

real :: xk, spline\_val, lagrange\_val

! Установка параметров интегрирования

a = 0.0

b = 20.0

relerr = 1.E-06

abserr = 0.0

h = 0.375

x = 1.0

x\_n = 1

DO WHILE (x <= 4)

x = x + h

x\_n = x\_n + 1

END DO

x = 1.0

ALLOCATE(x\_values(x\_n), f\_values(x\_n), b\_coef(x\_n), c\_coef(x\_n), d\_coef(x\_n))

i = 1

! Вычисление значений функции f(x) для 1 <= x <= 4 с шагом h

DO WHILE (x <= 4.0)

CALL quanc8(integral\_func, a, b, abserr, relerr, res, errest, nofun, flag)

x\_values(i) = x

f\_values(i) = res

x = x + h

i = i + 1

END DO

! Вызов подпрограммы SPLINE для вычисления коэффициентов сплайна

CALL SPLINE(x\_n, x\_values, f\_values, b\_coef, c\_coef, d\_coef)

! Сравнение значений в точках xk = 1.1875 + 0.375k (k=0,1,...,7)

WRITE(\*, '(A20, A20, A20, A20, A20, A20)') 'xk', 'f(xk)', 'spline(xk)', 'lagrange(xk)', 'f(xk)-spline(xk)', 'f(xk)-lagrange(xk)'

DO k = 0, 7

xk = 1.1875 + 0.375 \* k

! Вычисление значений сплайна, полинома Лагранжа и функции в точке xk

spline\_val = SEVAL(x\_n, xk, x\_values, f\_values, b\_coef, c\_coef, d\_coef)

lagrange\_val = compute\_lagrange(xk, x\_values, f\_values)

x = xk

CALL quanc8(integral\_func, a, b, abserr, relerr, res, errest, nofun, flag)

WRITE(\*, '(F20.4, F20.6, F20.6, F20.6, F20.6, F20.6)') xk,res,spline\_val,lagrange\_val,res-spline\_val,res-lagrange\_val

END DO

DEALLOCATE(x\_values, f\_values, b\_coef, c\_coef, d\_coef)

contains

REAL FUNCTION compute\_lagrange(x, x\_values, f\_values) RESULT(lagrange\_val)

REAL, INTENT(IN) :: x, x\_values(:), f\_values(:)

INTEGER :: n, i, j

REAL :: term, prod

n = SIZE(x\_values)

lagrange\_val = 0.0

DO i = 1, n

term = f\_values(i)

prod = 1.0

DO j = 1, n

IF (j /= i) THEN

prod = prod \* (x - x\_values(j)) / (x\_values(i) - x\_values(j))

END IF

END DO

lagrange\_val = lagrange\_val + term \* prod

END DO

END FUNCTION compute\_lagrange

end program main

# integral\_func.f90

MODULE integral\_func\_mod

IMPLICIT NONE

real :: x, h

CONTAINS

real FUNCTION integral\_func(z) RESULT(func\_value)

real, intent(in) :: z

func\_value = 1.0 / (EXP(z) \* (z + x))

END FUNCTION integral\_func

END MODULE integral\_func\_mod